# Error-Bounded Correction of Noisy Labels

**Songzhu Zheng** [1] **Pengxiang Wu** [2] **Aman Goswami** [3] **Mayank Goswami** [4] **Dimitris Metaxas** [2] **Chao Chen** [5]

## Abstract

To collect large scale annotated data, it is inevitable to introduce label noise, i.e., incorrect class labels. To be robust against label noise, many successful methods rely on the noisy classifiers (i.e., models trained on the noisy training data) to determine whether a label is trustworthy. However, it remains unknown why this heuristic works well in practice. In this paper, we provide the first theoretical explanation for these methods. We prove that the prediction of a noisy classifier can indeed be a good indicator of whether the label of a training data is clean. Based on the theoretical result, we propose a novel algorithm that corrects the labels based on the noisy classifier prediction. The corrected labels are consistent with the true Bayesian optimal classifier with high probability. We incorporate our label correction algorithm into the training of deep neural networks and train models that achieve superior testing performance on multiple public datasets.

## 1. Introduction

Label noise is ubiquitous in real world data. It may be caused by unintentional mistakes of manual or automatic annotators (Yan et al., 2014; Andreas et al., 2017). It may also be introduced by malicious attackers (Jacob et al., 2017). Noisy labels impair the performance of a model (Smyth et al., 1994; Brodley & Friedl, 1999), especially a deep neural network, which tends to have strong memorization power (Frénay & Verleysen, 2014; Zhang et al., 2017). Improving the robustness of a model trained with noisy labels is a crucial yet challenging task in many applications (Volodymyr & Geoffrey E., 2012; Wu et al., 2018).

Many methods have been proposed to train a robust model on data with label noise. One may re-calibrate the model by explicitly estimating a noise *transition matrix*, namely, the probability of one label being corrupted into another (Goldberger & Ben-Reuven, 2017; Patrini et al., 2017). One may also introduce hidden layers (Reed et al., 2014), prior on data distribution (Lee et al., 2019) or modified loss function (Van Rooyen et al., 2015; Shen & Sanghavi, 2019; Zhang & Sabuncu, 2018) to improve the robustness of the model. However, these methods either assume strong global priors on the data or lack sufficient supervision for the neural network to achieve satisfying performance. Furthermore, global model-correction mechanisms tend to rely on a few parameters; estimating these parameters can be challenging and the error will lead to failing of the training.

To adapt to heterogeneous noise pattern and to fully exploit the power of deep neural networks, *data-re-calibrating* methods have been proposed to focus on individual data instead of an overall model adjustment (Malach & Shalev-Shwartz, 2017; Jiang et al., 2018; Han et al., 2018; Tanaka et al., 2018; Wang et al., 2018; Ren et al., 2018; Cheng et al., 2020). These methods learn to re-calibrate the model on each individual datum depending on its own context. They gradually collect clean data whose labels are trustworthy. As more clean data are collected, the quality of the trained models improves. These methods slowly accumulate useful/trustworthy information and eventually attain state-of-the-art quality models.

Despite the success of data-re-calibrating methods, their underlying mechanism remains elusive. It is unclear why the neural nets trained on noisy labels can help select clean data. A theoretical underpinning will not only explain the phenomenon, but also advance the methodology. One major challenge for these methods is to control the data re-calibration quality. It is hard to monitor the model's re-calibrating decision on individual data. An aggressive selection of clean data can unknowingly accumulate irreversible errors. On the other hand, an overly-conservative strategy can be very slow in training, or stops with insufficient clean data and mediocre models. A theoretical guarantee will help develop models with self-assurance that the decision on each datum is reasonably close to the truth.

[1]Department of Applied Mathematics and Statistics, Stony Brook University, NY, USA [2]Department of Computer Science, Rutgers University, NJ, USA [3]Bain & Company, Bangalore, India. [4]Department of Computer Science, City University of New York, NY, USA [5]Department of Biomedical Informatics, Stony Brook University, NY, USA. Correspondence to: Songzhu Zheng <zheng.songzhu@stonybrook.edu>.

*Proceedings of the $37^{th}$ International Conference on Machine Learning*, Vienna, Austria, PMLR 119, 2020. 

In this paper, we provide *the first theoretical explanation for data-re-calibrating methods*. Our main theorem states that a noisy classifier (i.e., one trained on noisy labels) can identify whether a label has been corrupted. In particular, we prove that when the noisy classifier has low confidence on the label of a datum, such label is likely corrupted. In fact, we can quantify the *threshold of confidence*, below which the label is likely to be corrupted, and above which is it likely to be not. We also empirically show that the bound in our theorem is tight.

Our theoretical result not only explains existing data-re-calibrating methods, but also suggests a new solution for the problem. As a second contribution of this paper, we propose a novel method for noisy-labeled data. Based on our theorem and statistical principles, we verify the purity of a label through a likelihood ratio test w.r.t. the prediction of a noisy classifier, and the threshold value of confidence. The label is corrected or left intact depending on the test result. We prove that this simple label-correction algorithm has a guaranteed success rate and will recover the true labels with high probability. We incorporate the label-correction algorithm into the training of deep neural networks. We validate our method on different datasets with various noise patterns and levels. Our theoretically-founded method outperforms state-of-the-arts due to its simplicity and due to its principled design.

Our paper shows that a theorem that is well-grounded in applications will inspire elegant and powerful algorithms even in deep learning settings. Our contribution is two-fold:

- We provide a theorem quantifying how a noisy classifier's prediction correlates to the purity of a datum's label. This provides theoretical explanation for data-re-calibrating methods for noisy labels.
- Inspired by the theorem, we propose a new label-correction algorithm with guaranteed success rate. We train neural networks using the new algorithm and achieve superior performance.

The code of this paper can be found in `https://github.com/pingqingsheng/LRT.git`.

### 1.1. Related Work

One representative strategy for handling label noise is to model and employ noise transition matrix to correct the loss. For example, Patrini et al. (2017) propose to correct the loss function with estimated noise pattern. The resulting loss is an unbiased estimator of the ground truth loss, and enables the trained model to achieve better performance. However, such an estimator relies on strong assumptions and could be inaccurate in certain scenarios. Reed et al. (2014) consider modeling the noise pattern with a hidden layer. The learning of this hidden layer is regularized with a feature reconstruction loss, yet without a guarantee that the true label distribution is learned. Another method mentioned in their work is to minimize the entropy of neural network output; however, this method tends to predict a single class. To address this weakness, Dan et al. (2019) propose to utilize a small number of trusted, clean data to pre-train a network and estimate the noise pattern. However, such clean data may not always be available in practice.

Alternatively, another direction proposes to design models that are intrinsically robust to noisy data. Crammer et al. (2009) introduce a regularized confidence weighting learning algorithm (AROW), which attempts to preserve the weight distribution as much as possible while requiring the model to maintain discrimination ability. The follow-up work (Crammer & Lee 2010) improves this algorithm by herding the updating direction via specific velocity field (NHERD), and achieves better performance. Both of these works impose constraints on parameters, which, however, could prevent classifiers from adapting to complex datasets. Another similar strategy proposes to assume Gaussian distribution for features, and models the data with a robust generative classifier (Lee et al., 2019). However, such an assumption may not generalize to other complex scenarios.

Devansh et al. (2017) show that deep neural networks tend to learn meaningful patterns before they over-fit to noisy ones. Based on this observation, they propose to add Gaussian or adversarial noise to input when training with noisy labels, and empirically show that such data perturbation is able to make the resulting model more robust. Other commonly adopted techniques, such as weight decay and dropout, are also shown to be effective in increasing the robustness of trained classifier (Devansh et al. 2017; Zhang et al. 2017). However, the intrinsic reasons for this phenomenon still remain unclear and overfitting to noisy label is extremely likely. Data-re-calibrating methods select clean data while eliminating noisy ones during training. For example, Malach & Shalev-Shwartz (2017) and Han et al. (2018) train two networks simultaneously, and update the networks only with samples that are considered clean by both networks. Similarly, Jiang et al. (2018) also use two networks: the first one is pre-trained to learn a curriculum, and then utilized to select clean samples for training the second network. These methods deliver promising results but lack control of the quality of the collected clean data.

Finally, beyond deep learning framework, there are several theoretic works that demonstrate the robustness of a variety of losses to label noise (**?**; Nagarajan et al. 2013; **?**; Van Rooyen et al. 2015). Following the work of (Wang & Chaudhuri 2018), Gao et al. (2016) propose an algorithm that can converge to the Bayesian optimal classifier under different noise settings. Moreover, they provide in-depth discussion regarding the performance of k-nearest neighbor

(KNN) classifiers. However, the problem with KNN is that it is computationally intensive and difficult to be incorporated into a learning context. Within the framework of deep learning, there are more efforts that need to be made to bridge theory and practice.

## 2. The Main Theorem: Probing Label Purity Using the Noisy Classifier

Our main theorem answers the following question: without knowing the ground truth, how to decide whether a label is corrupted or not. During training, the only information one can rely on is a noisy classifier, i.e., one that is trained on the corrupted labels. Data-re-calibrating methods use the noisy classifier to decide whether a datum is clean-labeled. However, these methods lack a theoretical justification.

We establish the relationship between a noisy classifier and the purity of a label. We prove that if the classifier has low confidence on a datum with regard to its current label, then this label is likely corrupted. This result provides the first theoretical explanation of why noisy classifiers can be used to determine the purity of labels in previous methods.

This section is organized as follows. We start by providing basic notations and assumptions. Next, we state the main theorem for binary classification and then extend it to the multiclass setting. We also use experiments on synthetic data and CIFAR10 to validate the tightness of our bound.

### 2.1. Preliminaries and Assumptions

We first focus on binary classification. Later the result will be extended to multiclass setting. Let $\mathcal{X}$ be the feature space, $\mathcal{Y} = \{0, 1\}$ be the label space. The joint probability distribution, $D$, can be factored as $D(\boldsymbol{x}, y) = \Pr(y|\boldsymbol{x}) \Pr(\boldsymbol{x})$. We denote by $\eta(\boldsymbol{x}) = \Pr(y = 1|\boldsymbol{x})$ the *true conditional probability*. The *risk* of a binary classifier $h : \mathcal{X} \to \mathcal{Y}$ is $R(D, h) = \Pr_{(\boldsymbol{x},y)\sim D}[h(\boldsymbol{x}) \neq y]$. A *Bayes optimal classifier* is the minimizer of the risk over all possible hypotheses, i.e., $h^* = \arg\min_h R(D, h)$. It can be calculated using the true conditional probability, $\eta$,

$$h^*(\boldsymbol{x}) = \mathbf{1}_{\{\eta(\boldsymbol{x})>\frac{1}{2}\}}(\mathbf{x}) := \begin{cases} 1 & \eta(\boldsymbol{x}) > \frac{1}{2} \\ 0 & \text{otherwise} \end{cases}.$$

We assume $\eta$ satisfies the Tsybakov condition (Tsybakov, 2004). This condition, also called margin assumption, stipulates that the uncertainty of $\eta$ is bounded. In other words, the margin region close to the decision boundary, $\{\boldsymbol{x} \in \mathcal{X} \mid \eta(\boldsymbol{x}) = 1/2\}$, has a bounded volume.

**Assumption 1** (Tsybakov Condition)**.** *There exist constants $C, \lambda > 0$, and $t_0 \in (0, \frac{1}{2}]$, such that for all $t \leq t_0$,*

$$\Pr\left[\left|\eta(\boldsymbol{x}) - \frac{1}{2}\right| \leq t\right] \leq Ct^\lambda.$$

This assumption is adopted in previous works such as (Chaudhuri & Dasgupta, 2014; Belkin et al., 2018; Qiao et al., 2019). However, we have not seen any empirical verification of the condition in real datasets. In this paper, we conduct experiments to verify this condition and provide empirical estimation of the constants $C$ and $\lambda$. Our experiments indicate that this condition holds with moderate values of the constants $C$ and $\lambda$.

**The noisy label setting.** Instead of samples from $D$, we are given a sample set with noisy labels $\mathcal{S} = \{(\boldsymbol{x}, \widetilde{y})\}$, where $\widetilde{y}$ is the possibly corrupted label based on the true label $y$. We assume a *transition probability* $\tau_{i\to j} = \Pr(\widetilde{y} = j|y = i)$, i.e., the chance a true label $y$ is flipped from class $i$ to class $j$. For simplicity, we denote $\tau_{ij} = \tau_{i\to j}$. The transition probabilities $\tau_{01}$ and $\tau_{10}$ are independent of the true joint distribution $D$ and the feature $\boldsymbol{x}$. We denote the conditional probability of the noisy labels as $\widetilde{\eta}(\boldsymbol{x}) = \Pr(\widetilde{y} = 1|\boldsymbol{x})$. We call $\widetilde{\eta}$ the *noisy conditional probability*. It is easy to verify that $\widetilde{\eta}$ is linear to the true conditional probability, $\eta$:

$$\begin{aligned}\widetilde{\eta}(\boldsymbol{x}) &= (1 - \tau_{10})\eta(\boldsymbol{x}) + \tau_{01}[1 - \eta(\boldsymbol{x})] \\ &= (1 - \tau_{01} - \tau_{10})\eta(\boldsymbol{x}) + \tau_{01}.\end{aligned}$$

We intend to learn a classifier whose prediction is consistent with the Bayes optimal classifier $h^*$. Therefore, we call the prediction of $h^*$ the *correct label*.

**Definition 1** (Correct Label)**.** *Given $\boldsymbol{x}$, its* correct label *is the Bayes optimal classifier prediction $h^*(\boldsymbol{x})$.*

The correct label, $h^*(\boldsymbol{x})$, is subtly different from the true label, $y$. In particular, $h^*(\boldsymbol{x})$ is uniquely decided by $\eta(\boldsymbol{x})$, whereas $y$ is a sample from $\eta$. Since $h^*$ is our final goal, we focus on recovering the correct label, $h^*(\boldsymbol{x})$, instead of $y$.

### 2.2. The Main Theorem

Our main theorem connects a noisy classifier $f : \mathcal{X} \to \mathcal{Y}$ with the chance of a noisy label $\widetilde{y}$ being correct. We assume $f$ is trained on the noisy labels and is trained well enough, i.e., $\epsilon$-close to the noisy conditional probability, $\widetilde{\eta}$. For convenience, we denote by $f_{\widetilde{y}}$ the classifier prediction of label being $\widetilde{y}$, formally, $f_{\widetilde{y}}(\boldsymbol{x}) = f(\boldsymbol{x})$ if $\widetilde{y} = 1$, and $1 - f(\boldsymbol{x})$ otherwise. Define the estimation error $\epsilon := \|f - \widetilde{\eta}\|_\infty$.

**Theorem 1.** *Assume $\eta(\boldsymbol{x})$ satisfies the Tsybakov condition with constants $C, \lambda > 0$, and $t_0 \in (0, \frac{1}{2}]$. Assume $\epsilon \leq t_0(1 - \tau_{10} - \tau_{01})$. For $\Delta = \frac{1-|\tau_{10}-\tau_{01}|}{2}$, we have:*

$$\Pr_{(x,y)\sim D}\left[\widetilde{y} = h^*(\boldsymbol{x}), f_{\widetilde{y}}(\boldsymbol{x}) < \Delta\right] \leq C\Big[O(\epsilon)\Big]^\lambda.$$

**Implication of the theorem.** Intuitively, the theorem states that a noisy label $\widetilde{y}$ has bounded probability to be correct if it has a low vote-of-confidence by $f$. The upper bound of

the probability is controlled by $\epsilon$, the approximation error of $f$. In other words, the better $f$ approximates $\widetilde{\eta}$, the tighter the bound is. This justifies the usage of a good-quality $f$ to determine if $\widetilde{y}$ is trustworthy. Later we will show $\epsilon$ is reasonably small in deep learning setting and the bound is tight in practice.

We remark that the constant $\Delta$ and the constant hidden inside the big-O in the theorem depend on $\tau_{ij}$'s, which are unknown in practice. Based on this theorem, we will propose a new label-correction algorithm that determines $\Delta$ robustly in practice without knowing $\tau_{ij}$'s.

#### 2.2.1. Proof of Theorem 1

**Preliminary Lemmata.** To prove this theorem, we need to first prove two lemmata. Lemma 1 will show that if a classifier $g$ is a linear transformation of $\eta$, when the value $g_{\widetilde{y}}$ is below a certain threshold, $\widetilde{y}$ is unlikely to be consistent with the true Bayesian optimal decision, $h^*$. Next, Lemma 2 states that since $\widetilde{\eta}(\boldsymbol{x})$ is a linear transformation of $\eta(\boldsymbol{x})$, Lemma 1 will apply to $\widetilde{\eta}(\boldsymbol{x})$ and $\Delta$ can be set accordingly. Finally, based on the conclusion of Lemma 2 and the Tsybakov condition, we can upperbound $\Pr\left[\widetilde{y}=h^*(\boldsymbol{x}), f_{\widetilde{y}}(\boldsymbol{x})<\Delta\right]$ if $f$ is $\epsilon$-close to $\widetilde{\eta}$.

**Lemma 1.** *If a classifier $g$ depends linearly on $\eta$, i.e., $g(\boldsymbol{x}) = a\eta(\boldsymbol{x}) + b$ with $a, b > 0$. Set $\Delta = \min\left(\frac{a}{2}+b, 1-b-\frac{a}{2}\right)$. We have*

$$\Pr_{(x,y)\sim D}\left[\widetilde{y}=h^*(\boldsymbol{x}), g_{\widetilde{y}}(\boldsymbol{x})<\Delta\right]=0 \tag{1}$$

*Proof.* To calculate $\Pr_{(x,y)\sim D}\left[\widetilde{y}=h^*(\boldsymbol{x}), g_{\widetilde{y}}(\boldsymbol{x})<\Delta\right]$, we enumerate two cases:

Case 1: $\widetilde{y}=1$. Observe $h^*(\boldsymbol{x})=1$ iff $\eta(\boldsymbol{x})>1/2$; $g_{\widetilde{y}}(\boldsymbol{x}) = g(\boldsymbol{x}) = a\eta(\boldsymbol{x})+b < \Delta$ iff $\eta(\boldsymbol{x}) < \frac{\Delta-b}{a}$. We have:

$$\Pr\left[\widetilde{y}=h^*(\boldsymbol{x}), g_{\widetilde{y}}(\boldsymbol{x})<\Delta\right] = \Pr\left[\frac{1}{2}<\eta(\boldsymbol{x})<\frac{\Delta-b}{a}\right]. \tag{2}$$

We next show that this probability is 0 for the chosen $\Delta = \min\left(\frac{a}{2}+b, 1-b-\frac{a}{2}\right)$. If $\Delta=\frac{a}{2}+b$, the probability is zero as $\frac{\Delta-b}{a}=\frac{1}{2}$. Otherwise, $\Delta = 1-b-\frac{a}{2}$. We know that $1-b-\frac{a}{2}<\frac{a}{2}+b$. Therefore, $1-2b<a$. In this case,

$$\frac{\Delta-b}{a}=\frac{1-2b}{a}-\frac{1}{2}<1-\frac{1}{2}=\frac{1}{2}.$$

Thus we have $\Pr\left[\frac{1}{2}<\eta(\boldsymbol{x})<\frac{\Delta-b}{a}\right]=0$.

Case 2: $\widetilde{y}=0$. Observe that $h^*(\boldsymbol{x})=0$ iff $\eta(\boldsymbol{x})\le 1/2$; $g_{\widetilde{y}}(\boldsymbol{x}) = 1-g(\boldsymbol{x}) = 1-[a\eta(\boldsymbol{x})+b] < \Delta$ iff $\eta(\boldsymbol{x}) > L := \frac{1-b-\Delta}{a}$, we have:

$$\Pr\left[\widetilde{y}=h^*(\boldsymbol{x}), g_{\widetilde{y}}(\boldsymbol{x})<\Delta\right] = \Pr\left[L<\eta(\boldsymbol{x})<\frac{1}{2}\right].$$

Similar to Case 1, by checking when $\Delta=\frac{a}{2}+b$ and when $\Delta = 1-b-\frac{a}{2}$, we can verify that $\Pr\left[\frac{1-b-\Delta}{a}<\eta(\boldsymbol{x})<\frac{1}{2}\right]=0$.

This proves Equation (1) and completes the proof. □

**Lemma 2.** *Let $\Delta = \frac{1-|\tau_{10}-\tau_{01}|}{2}$. Let $\widetilde{\eta}_1 = \widetilde{\eta}$ and $\widetilde{\eta}_0 = 1-\widetilde{\eta}$.*

$$\Pr_{(x,y)\sim D}\left[\widetilde{y}=h^*(\boldsymbol{x}), \widetilde{\eta}_{\widetilde{y}}(\boldsymbol{x})<\Delta\right]=0. \tag{3}$$

*Proof.* Recall $\widetilde{\eta}(\boldsymbol{x}) = (1-\tau_{01}-\tau_{10})\eta(\boldsymbol{x})+\tau_{01}$, in which $\tau_{01}$ and $\tau_{10}$ are transition probabilities. We can directly prove this lemma using Lemma 1 by setting $g=\widetilde{\eta}$ with $a=1-\tau_{01}-\tau_{10}$ and $b=\tau_{01}$. □

**Proof of Theorem 1 using the Lemmata.**

*Proof.* When $\widetilde{y}=1$, $f_{\widetilde{y}}(\boldsymbol{x}) = f(\boldsymbol{x}) \ge \widetilde{\eta}(\boldsymbol{x})-\epsilon$.

$$\Pr\left[\widetilde{y}=h^*(\boldsymbol{x}), f_{\widetilde{y}}(\boldsymbol{x})<\Delta\right] \le \Pr\left[\widetilde{y}=h^*(\boldsymbol{x}), \widetilde{\eta}(\boldsymbol{x})-\epsilon<\Delta\right]$$

Substituting $\Delta$ with $\Delta+\epsilon$ into equation (2), we have:

$$\begin{aligned}
&\Pr\left[\widetilde{y}=h^*(\boldsymbol{x})=1, \widetilde{\eta}(\boldsymbol{x})-\epsilon<\Delta\right]\\
=&\Pr\left[\widetilde{y}=h^*(\boldsymbol{x})=1, \widetilde{\eta}(\boldsymbol{x})<\Delta+\epsilon\right]\\
=&\Pr\left[\frac{1}{2}<\eta(\boldsymbol{x})<\frac{\Delta+\epsilon-\tau_{01}}{1-\tau_{01}-\tau_{10}}\right]
\end{aligned}$$

Similar to Lemma 1, by discussing the cases when $\Delta=\frac{1+\tau_{10}-\tau_{01}}{2}$ and when $\Delta=\frac{1+\tau_{01}-\tau_{10}}{2}$, we can show that $\frac{\Delta-\tau_{01}}{1-\tau_{01}-\tau_{10}}<\frac{1}{2}$. Based on the Tsybakov condition, we have

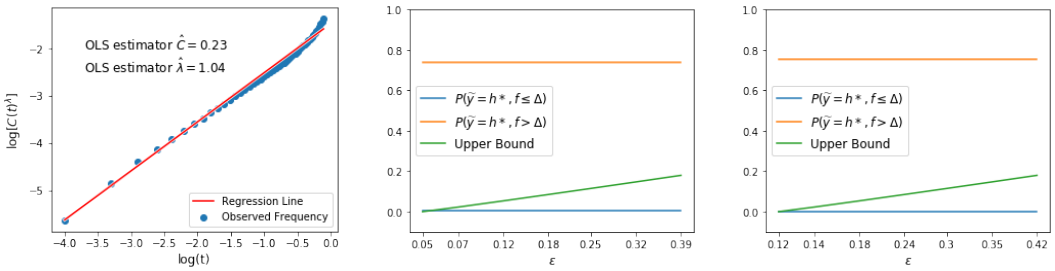
$$\begin{aligned}
&\Pr\left[\frac{1}{2}<\eta(\boldsymbol{x})<\frac{\Delta-\tau_{01}}{1-\tau_{01}-\tau_{10}}+\frac{\epsilon}{1-\tau_{01}-\tau_{10}}\right]\\
\le&\Pr\left[\frac{1}{2}<\eta(\boldsymbol{x})<\frac{1}{2}+\frac{\epsilon}{1-\tau_{01}-\tau_{10}}\right] \le C\left(\frac{\epsilon}{1-\tau_{01}-\tau_{10}}\right)^{\lambda}
\end{aligned}$$

This implies that:

$$\Pr\left[\widetilde{y}=h^*(\boldsymbol{x})=1, f_{\widetilde{y}}(\boldsymbol{x})<\Delta\right] \le C\left(\frac{\epsilon}{1-\tau_{01}-\tau_{10}}\right)^{\lambda}$$

Similar to case 1 of Lemma 1, by using equation (3) for the case when $\widetilde{y}=0$, we can prove that

$$\begin{aligned}
&\Pr\left[\widetilde{y}=h^*(\boldsymbol{x})=0, f_{\widetilde{y}(\boldsymbol{x})}<\Delta\right]\\
\le&\Pr\left[\widetilde{y}=h^*(\boldsymbol{x})=0, 1-\widetilde{\eta}(\boldsymbol{x})-\epsilon<\Delta\right]\\
=&\Pr\left[\frac{1-\tau_{01}-\Delta}{1-\tau_{10}-\tau_{01}}-\frac{\epsilon}{1-\tau_{10}-\tau_{01}}<\eta(\boldsymbol{x})<\frac{1}{2}\right]\\
\le&\Pr\left[\frac{1}{2}-\frac{\epsilon}{1-\tau_{01}-\tau_{10}}<\eta(\boldsymbol{x})<\frac{1}{2}\right]\\
\le&C\left(\frac{\epsilon}{1-\tau_{01}-\tau_{10}}\right)^{\lambda}
\end{aligned}$$




(a) (b) (c)

*Figure 1.* Synthetic experiment using CIFAR10 at noise level 20%. (a): Check of Tsybakov condition using linear regression. Where y-axis is the proportion of data points at distance $t$ from decision boundary. (b): Proportion of labels that are not correct (not consistent with Bayes optimal decision rule) and the proposed upper bound. (c). Same as (b) but labels are corrupted with asymmetric noise.

Combining the two cases ($\widetilde{y} = 1$) and ($\widetilde{y} = 0$) completes the proof. □

**Remark 1.** *Indeed, we can also prove a bound for the opposite case: when $f_{\widetilde{y}}$ is highly confident, $\widetilde{y}$ is correct with high probability. In this paper, we only focus on the bound in theorem 1 as we only want to identify incorrect labels and fix them.*

### 2.3. Multiclass Setting

Theorem 1 can be generalized to a multiclass setting. Let $\widetilde{y}$ be the observed (possibly) corrupted label, $\eta_i(\boldsymbol{x}) = \Pr(y = i \mid \boldsymbol{x})$ and $\widetilde{\eta}_i(\boldsymbol{x}) = \Pr(\widetilde{y} = i \mid \boldsymbol{x})$. Recall $f_i(\boldsymbol{x})$ is the classifier's prediction on label $i$. Define $N_c$ to be the number of total classes and $[N_c] = \{1, 2, \cdots, N_c\}$.

First we extend the Tsybakov condition to multiclass scenario (Chen & Sun, 2006). Denote by $u_{\boldsymbol{x}}$ the Bayes optimal classifier prediction, or say the class predicted by $\eta(\boldsymbol{x})$, formally $u_{\boldsymbol{x}} := h^*(\boldsymbol{x}) = \arg\max_i \eta_i(\boldsymbol{x})$. Denote by $s_{\boldsymbol{x}}$ the second best prediction, $s_{\boldsymbol{x}} := \arg\max_{i \neq u_{\boldsymbol{x}}} \eta_i(\boldsymbol{x})$. The difference between their corresponding true conditional probability is a non-negative function, whose zero level set $\{\boldsymbol{x} | \eta_{u_{\boldsymbol{x}}}(\boldsymbol{x}) - \eta_{s_{\boldsymbol{x}}}(\boldsymbol{x}) = 0\}$ is the decision boundary of $h^*$. We assume the Tsybakov condition around the margin of this decision boundary: $\exists C, \lambda > 0$ and $\exists t_0 \in (0, 1]$, such that for all $t \leq t_0$,

$$\Pr\left[\eta_{u_{\boldsymbol{x}}}(\boldsymbol{x}) - \eta_{s_{\boldsymbol{x}}}(\boldsymbol{x}) \leq t\right] \leq C t^{\lambda} \tag{4}$$

For any pair of labels $i, j \in [N_c]$, we have the linear relationship $\widetilde{\eta}_i(\boldsymbol{x}) = \sum_{j \in [N_c]} \tau_{ji} \eta_j(\boldsymbol{x})$. Define $m_{\boldsymbol{x}} := \arg\max_i f_i(\boldsymbol{x})$. Define the estimation error $\epsilon := \max_{\boldsymbol{x}, i} |f_i(\boldsymbol{x}) - \widetilde{\eta}_i(\boldsymbol{x})|$.

**Theorem 2.** *Assume $\eta(\boldsymbol{x})$ fulfills multi-class Tsybakov condition for constants $C, \lambda > 0$ and $t_0 \in (0, 1]$. Assume that $\epsilon \leq t_0 \min_i \tau_{i,i}$. For $\Delta = \min\left[1, \min_{\boldsymbol{x}} [\tau_{\widetilde{y},\widetilde{y}} \eta_{s_{\boldsymbol{x}}}(\boldsymbol{x}) + \sum_{j \neq \widetilde{y}} \tau_{j,\widetilde{y}} \eta_j(\boldsymbol{x})]\right]$:*

$$\Pr_{(x,y) \sim D}\left[\widetilde{y} = h^*(\boldsymbol{x}), f_{\widetilde{y}}(\boldsymbol{x}) < \Delta\right] \leq C\left[O(\epsilon)\right]^{\lambda}$$

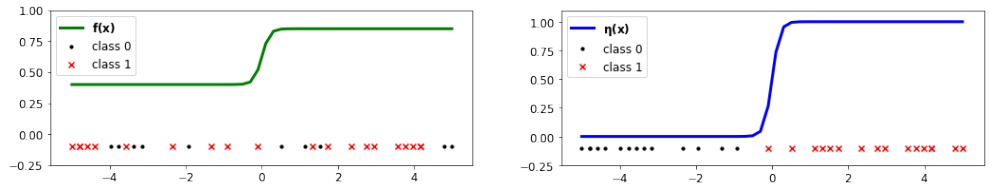
The proof of Theorem 2 will be provided in supplementary material.

### 2.4. Empirical Validation of the Bound

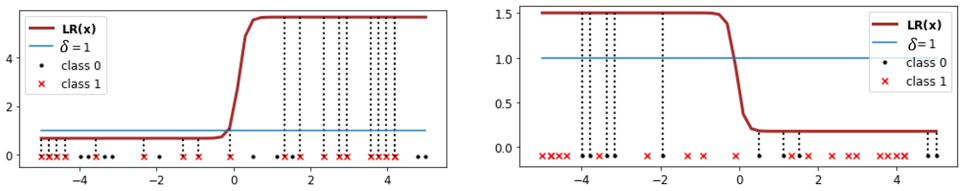
To better understand the Tsybakov condition assumption and the bound in our theorem, we conduct the following experiment. On the CIFAR10 dataset, we train deep neural networks to approximate relevant functions. We use these functions to estimate the constants $C$ and $\lambda$ in the Tsybakov condition. Using these constants, we calculate the bound in Theorem 2 as a function of $\epsilon$ and check if it is tight.

To estimate $C$ and $\lambda$, we approximate the true conditional probability $\eta$ using a deep neural network trained on the original clean-labeled CIFAR10 data. We densely sample $t$ between 0 and 0.9. For each $t$, we empirically evaluate the left hand side (LHS) probability of Equation (4) and then use these values to estimate $C$ and $\lambda$ via regression. In particular, for each $t$ we calculate LHS of Equation (4) using the frequency $p_t = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{\eta_{m_{\boldsymbol{x}}}(\boldsymbol{x}) - \eta_{s_{\boldsymbol{x}}}(\boldsymbol{x}) \leq t\}}(\boldsymbol{x})$, in which $n$ is the number of data. If the RHS bound is tight, we can use $\log p_t$ to approximate $\log(Ct^{\lambda})$. $\log(Ct^{\lambda}) = \log C + \lambda \log t$. As shown in Figure 1(a), we plot all $(\log t, \log(Ct^{\lambda}))$ pairs as blue dots and estimate $C$ and $\lambda$ via linear regression (red line). We observe that the samples are quite close to linear. Indeed, we could get ordinary least square (OLS) estimator of constant $C$ and $\lambda$ with high confidence (determinant coefficient $R^2 = 0.99$, p-value $< 1e - 53$). The estimated $C$ and $\lambda$ are 0.23 and 1.04 respectively.

Next, we verify our bound in Theorem 2. Using the estimated $C$ and $\lambda$, we can calculate the bound (RHS of Equation (4)) as a function of $\epsilon$ (the constant in the big-O is




(a) Noisy labels and $f$. (b) Corrected labels and $\eta$.




(c) LR for $\widetilde{y} = 1$. (d) LR for $\widetilde{y} = 0$.

*Figure 2.* An illustration of the label correction algorithm. $\delta$ is set to 1. (a): a corrupted sample and its corresponding classifier prediction $f$. (b): after correction, the labels are consistent with the true conditional probability, $\eta$. (c): likelihood ratio for $\widetilde{y} = 1$. Data with $x < 0$ are corrected to $\widetilde{\eta}_{new} = 0$ as $LR(\boldsymbol{x})$ are below $\delta = 1$. (d): likelihood ratio for $\widetilde{y} = 0$. Data with $x > 0$ are corrected to $\widetilde{\eta}_{new} = 1$ as $LR(\boldsymbol{x})$ are below $\delta = 1$.

provided in the supplemental material). In Figure 1(b), we plot the bound function in green curve. We compare this bound with the LHS of Equation (4) which we can empirically evaluate. In particular, we train a noisy classifier $f$ by training a neural network on noisy labels (symmetric noise level 20%, see Section 4 for details). Using $f$, we can count the number of data points which has $f_{\widetilde{y}} \leq \Delta$ and meanwhile $\widetilde{y}$ is equal to $h^*(\boldsymbol{x})$ (calculated using $\eta$: the clean-label-trained neural network). This gives us the LHS of Equation (4), which is the probability of a label being correct when $f$ has low confidence (blue line in Figure 1(b)). Similarly, we can calculate the probability of a label being correct when $f$ has high confidence (orange line in Fig. 1(b)). We also carry out the same experiment on a different noise setting (asymmetric noise level 20%, see Sec. 4 for details).

**Discussion.** On CIFAR10 dataset, we estimated the constants of Tsybakov condition to be $C = 0.23$ and $\lambda = 1.04$ with high confidence. This means our bound (Equation (4) is almost linear. As observed in Figure 1(b) and (c), the bound is rather small (only up to 0.2 when the approximation error of the classifier, $\epsilon$, is below 0.4). Furthermore, the empirically evaluated chance of $\widetilde{y}$ being correct when $f$ has low confidence (blue lines Figure 1) is almost zero, well below the curve of the bound. In Figure 1(b) The fact that the blue and green line intersects at $\epsilon = 0.06$ implies that $\epsilon$ can be as small as 0.06. Similarly, Figure 1(c) implies $\epsilon$ can be as small as 0.12. Finally, we note that the orange lines are well above the blue ones. This means when $f$ has high confidence on $\widetilde{y}$, there is a high chance $\widetilde{y}$ is correct. In other words, by comparing $f_{\widetilde{y}}$ with a properly chosen constant $\Delta$, we can identify most data with corrupted labels.

We also conduct experiments on synthetic data (generated using multivariate normal distribution). In such case, we can calculate $\eta$ and $\widetilde{\eta}$ exactly. The estimated $C$ and $\lambda$ are 0.6 and 1.3 respectively. More details about the synthetic experiments can be found in the supplemental material.

In conclusion, experiments on synthetic and on CIFAR10 datasets show that the constants in Tsybakov condition are rather small and the bound in our theorem is almost linear to $\epsilon$. We also note the bound is generally small/tight even in deep learning setting. Thresholding $f$'s confidence does detect corrupted labels accurately.

## 3. The Algorithm: Likelihood Ratio Test for Label Correction

Our theoretical insight inspires a new algorithm for label correction. We propose to directly test the confidence level of the noisy classifier to determine whether a label is correct. One additional requirement is that if we decide that a label is incorrect, we also need to decide what is the correct label. Therefore, instead of checking the confidence level, we check the likelihood ratio between $f$'s confidence on $\widetilde{y}$ and its confidence on its own label prediction, i.e., $m_{\boldsymbol{x}}$. Specifically, we check the likelihood ratio

$$\mathrm{LR}(f, \boldsymbol{x}, \widetilde{y}) = f_{\widetilde{y}}(\boldsymbol{x}) / f_{m_{\boldsymbol{x}}}(\boldsymbol{x}).$$

We compare this likelihood ratio with a predetermined threshold $\delta$. The value of $\delta$ is given in the next theorem. This is essentially a hypothesis testing on the null hypothesis $H_0 : \widetilde{y} = h^*(\boldsymbol{x})$. If $\mathrm{LR}(f, \boldsymbol{x}, y) < \delta$, we reject the null hypothesis and flip the label $\widetilde{y}_{new} = m_{\boldsymbol{x}}$. Otherwise, the label remains unchanged, $\widetilde{y}_{new} = \widetilde{y}$. If $\widetilde{y} = m_{\boldsymbol{x}}$ then the likelihood ratio is 1, $\widetilde{y}_{new} = m_{\boldsymbol{x}} = \widetilde{y}$. Detailed algorithm

is provided in Procedure 1. See Figure 2 for an illustration of the algorithm in a binary classification case.

**Procedure 1** `LRT-Correction`

**Input:** $(\boldsymbol{x}, \widetilde{y}), f(\boldsymbol{x}), \delta$.
**Output:** $\widetilde{y}_{new}$

$m_{\boldsymbol{x}} := \arg\max_i f_i(\boldsymbol{x})$
$\mathrm{LR}(f, \boldsymbol{x}, \widetilde{y}) := f_{\widetilde{y}}(\boldsymbol{x}) / f_{m_{\boldsymbol{x}}}(\boldsymbol{x})$
**if** $\mathrm{LR}(f, \boldsymbol{x}, \widetilde{y}) < \delta$ **then**
  $\widetilde{y}_{new} = m_{\boldsymbol{x}}$
**else**
  $\widetilde{y}_{new} = \widetilde{y}$
**end if**

We will show in the following theorem that the LRT correction algorithm is guaranteed to make proper correction and clean most of the corrupted labels. In particular, we show that in practice if we have a reasonable approximation $\hat{\delta}$ to the theoretically optimal $\delta$, the algorithm flips $\widetilde{y}$ to the correct label (the Bayes optimal prediction, $h^*(\boldsymbol{x})$) with a good chance. Recall the approximation error of the classifier is $\epsilon := \max_{\boldsymbol{x},i} |f_i(\boldsymbol{x}) - \widetilde{\eta}_i(\boldsymbol{x})|$.

We consider two cases: (1) the label being flipped $y_{new} = m_{\boldsymbol{x}}$; and (2) the label remaining the same $y_{new} = \widetilde{y}$. Each case has its own ideal $\delta$. We bound the probability of obtaining a correct label with $\epsilon$ and $\xi$. Here $\xi$ is the difference between the chosen $\hat{\delta}$ and the ideal $\delta$. We also introduce an additional term, $\Psi$, denoting the probability that the true label is neither $\widetilde{y}$ nor $m_{\boldsymbol{x}}$, formally, $\Psi = \Pr_{(\boldsymbol{x},y)\sim D}[u_{\boldsymbol{x}} \notin \{m_{\boldsymbol{x}}, \widetilde{y}\}]$.

**Theorem 3.** *$\forall i, j \in [N_c]$, assume $\eta(\boldsymbol{x})$ fulfills multi-class Tsybakov condition for constants $C > 0$, $\lambda > 0$, $t_0 \in (0, 1]$.*

*Case 1 (Label flipped by* `LRT-Corr` *$((\boldsymbol{x}, \widetilde{y}), f(\boldsymbol{x}), \hat{\delta})$):*
*let* $\delta_1 = \min_{\boldsymbol{x}} \left[ \frac{1}{f_{m_{\boldsymbol{x}}}(\boldsymbol{x})} \left( \tau_{\widetilde{y},\widetilde{y}} \eta_{s_{\boldsymbol{x}}}(\boldsymbol{x}) + \sum_{j \neq \widetilde{y}} \tau_{j,\widetilde{y}} \eta_j(\boldsymbol{x}) \right) \right]$ *and* $\xi_1 := |\hat{\delta} - \delta_1|$. *Assume* $\xi_1 \leq \delta_1$ *and* $\epsilon \leq \min\left( \frac{t_0 \delta_1^2 \min_i \tau_{ii} - \xi_1^2 - \xi_1}{\delta_1^2}, (t_0 - \xi_1) \min_i \tau_{ii} \right)$.
*Then:* $\Pr_{(x,y)\sim D}[\widetilde{y}_{new} = h^*(\boldsymbol{x}), \widetilde{y}$ *is flipped* $]$ *is at least* $1 - C\left[O(\max(\epsilon, \xi_1))\right]^{\lambda} - \Psi$.

*Case 2 (Label preserved by* `LRT-Corr` *$((\boldsymbol{x}, \widetilde{y}), f(\boldsymbol{x}), \hat{\delta})$):*
*let* $\delta_2 = \max_{\boldsymbol{x}} \left[ \frac{f_{\widetilde{y}}(\boldsymbol{x})}{\tau_{m_{\boldsymbol{x}},m_{\boldsymbol{x}}} \eta_{s_{\boldsymbol{x}}}(\boldsymbol{x}) + \sum_{j \neq m_{\boldsymbol{x}}} \tau_{j,m_{\boldsymbol{x}}} \eta_j(\boldsymbol{x})} \right]$ *and* $\xi_2 := |\hat{\delta} - \delta_2|$. *Assume* $\xi_2 \leq \delta_2$ *and* $\epsilon \leq \min\left( \frac{t_0 \delta_2^2 \min_i \tau_{ii} - \xi_2^2 - \xi_2}{\delta_2^2}, (t_0 - \xi_2) \min_i \tau_{ii} \right)$.
*Then:* $\Pr_{(x,y)\sim D}[\widetilde{y}_{new} = h^*(\boldsymbol{x}),\ \widetilde{y}$ *isn't flipped*$]$ *is at least* $1 - C\left[O(\max(\epsilon, \xi))\right]^{\lambda} - \Psi$.

### 3.1. Training Deep Nets with LRT-Correction

We incorporate the proposed label-correction into the training of deep neural networks. Similar to other data-recalibrating methods, our training algorithm continuously trains a deep neural network while correcting the noisy labels. Procedure 2 is the pseudocode of the training method, called `AdaCorr`. It trains a neural network model iteratively. Each iteration includes both label correction and model training steps. In label correction step, the prediction of the current neural network, $f$, is used to run LRT test on all training data, and to correct their labels according to the test result. Since $f$ is used to approximate the conditional probability $\widetilde{\eta}$, we use the softmax layer output of the neural network as $f$. After the labels of all training data are updated, we use them to train the neural network incrementally. We continue this iterative procedure until the training converges.

We also have a burn-in stage in which we train the network using the original noisy labels for $m$ epochs. During the burn-in stage, we use the original cross-entropy loss, $L_{CE}$. Afterwards, we add an additional retroactive loss, with the intention of stabilizing the network and avoiding overfitting.

After the burn-in stage, we want to avoid overfitting of the neural network, so that its output better approximates $\widetilde{\eta}$. To achieve this goal, we introduce a *retroactive loss* term $L_{retro}(f(\boldsymbol{x}), \widetilde{y})$. The idea is to enforce the consistency between $f$ and the prediction of the model at a previous epoch, $f^r$. It has been observed that a neural network at earlier training stage tends to learn the true pattern rather than to overfit the noise (Devansh et al., 2017). Formally, the loss can be written as $\sum_{c=1}^{N_c} f_c^r(\boldsymbol{x}) \log f_c(\boldsymbol{x})$, in which $N_c$ is the number of possible label classes. The training loss is the sum of the retroactive loss and the cross-entropy loss:

$$\begin{aligned} L(f(\boldsymbol{x}), \widetilde{y}, f^r) &= L_{retro}(f(\boldsymbol{x}), f^r(\boldsymbol{x})) + L_{CE}(f(\boldsymbol{x}), \widetilde{y}) \\ &= \sum_{c=1}^{N_c} f_c^r(\boldsymbol{x}) \log f_c(\boldsymbol{x}) + \sum_{c=1}^{N_c} \widetilde{y}_c \log f_c(\boldsymbol{x}). \end{aligned}$$

**Procedure 2** `AdaCorr`

**Input:** $S = \{(\boldsymbol{x}, \widetilde{y})\}, \delta, m, T$

**for** epoch=1 to $m$ **do**
  Train neural network with $L_{CE}$
**end for**
$f^r =$ current model prediction
**for** epoch=$m+1$ to $T$ **do**
  **if** epoch $\geq m + 10$ **then**
    $f =$ current model prediction
    **for all** $(\boldsymbol{x}, \widetilde{y}) \in S$ **do**
      $\widetilde{y}_{new}$= `LRT-Correction`$((\boldsymbol{x}, \widetilde{y}), f, \delta)$
      $\widetilde{y} = \widetilde{y}_{new}$
    **end for**
  **end if**
  Train using $L_{retro} + L_{CE}$, with $f^r$ and $\widetilde{y}$
**end for**

In the experiment we evaluate our method on 4 public

datasets: CIFAR10, CIFAR100, MNIST and ModelNet40 (see Section 4 for more details). Based on previous observations (Devansh et al., 2017), on CIFAR10 and CIFAR100 datasets, a neural network takes about 30 epochs to fit the true pattern before overfitting the noise. We use this number as the burn-in stage length $m$. For easier datasets like MNIST and ModelNet40, we set $m$ to be slightly smaller (25). As for $\delta$, setting $\delta$ to be slightly smaller than 1 seems sufficient. Our Theorem 3 guarantees that the bound is affected almost linearly (as $\lambda \approx 1$ per Section 2.4) to the error of the manually picked $\delta$ from the optimal one.

## 4. Experiments

In this section we empirically evaluate our proposed method with several datasets, where noisy labels are injected according to specified noise transition matrices.

**Datasets.** We use the following datasets: MNIST (LeCun et al. 1998), CIFAR10 (Krizhevsky et al. 2009), CIFAR100 (Krizhevsky et al. 2009) and ModelNet40 (Wu et al. 2015). MNIST consists of $28 \times 28$ grayscale images with 10 categories. It contains 60,000 images, and we use 45,000 for training, 5,000 for validation and 10,000 for testing. CIFAR10 and CIFAR100 consist of the same 60,000 images whose size is $32 \times 32 \times 3$. CIFAR10 has 10 classes while CIFAR100 has 100 fine-grained classes. Similar to MNIST, we split 90% and 10% data from the official training set for the training and validation, respectively, and use the official test set for testing. ModelNet40 contains 12,311 CAD models from 40 categories, where 8,859 are used for training, 984 for validation and the remaining 2,468 for testing. We follow the protocol of (Qi et al., 2017) to convert the CAD models into point clouds by uniformly sampling 1,024 points from the triangular mesh and normalizing them within a unit ball. In all experiments, we use early stopping on validation set to tune hyperparameters and report the performance on test set.

**Baselines.** We compare the proposed method with the following methods: (1) *Standard*, which trains the network in a standard manner, without any label resistance technique; (2) *Forward Correction* (Patrini et al. 2017), which explicitly estimates the noise transition matrix to correct the training loss; (3) *Decoupling* (Malach & Shalev-Shwartz 2017), which trains two networks simultaneously and updates the parameters on selected data whose labels are possibly clean; (4) *Coteaching* (Han et al. 2018), which also trains two networks but exchanges their error information for network updating; (5) *MentorNet* (Jiang et al. 2018), which learns a curriculum to filter out noisy data; (6) *Forgetting* (Devansh et al., 2017), which uses dropout to help deep models resist label noise. (7) *Abstention* (Thulasidasan et al. 2019), which regularizes the network with abstention loss to ensure model robustness under label noise.

**Experimental setup.** For the classification of MNIST, CIFAR10 and CIFAR100, we use preactive ResNet-34 (He et al. 2016) as the backbone for all the methods. On ModelNet40, we use PointNet. We train the models for 180 epochs to ensure that all the methods have converged. We utilize RAdam (Liu et al. 2019) for the network optimization, and adopt batch size 128 for all the datasets. We use an initial learning rate of 0.001, which is decayed by 0.5 very 60 epochs. We also update $f^r$ to $f$ once at epoch $m + 40$ to reflect better predictive power of network after several epochs. The experimental results are listed in Table 2. As is shown, our method overall achieves the best performance across the datasets under different noise settings.

**Clothing 1M.** We also evaluate our method on a large scale Clothing 1M dataset (Xiao et al., 2015), which consists of 1M images with real-world noisy labels. We use pre-trained ResNet-50 and train the model using SGD for 20 epochs. Our method achieves accuracy 71.47%. It outperforms Standard (68.94%), Forward Correction (69.84%) and Backward Correction (Patrini et al., 2017) (69.13%), where we take the number from the original paper directly. Note that other baselines (Forgetting, Decoupling, MentorNet, Coteaching and Abstention) did not report results on this dataset.

*Table 1.* Performance on Clothing 1M Dataset

| Method | Accuracy(%) |
|---|---|
| Standard | 68.94 |
| Forward | 69.84 |
| Backward | 69.13 |
| AdaCorr | $71.74 \pm 0.12$ |

**Discussion.** Our method outperform state-of-the-arts over a broad spectrum of noise patterns and levels. This is due to the relatively simple procedure our theoretically guaranteed algorithm. Looking closely, in Figure 3, we draw convergence curves on CIFAR10 with 0.4 uniform noise. On the left, we show the curves of our proposed AdaCorr method. The model continues to flip labels to correct ones. Meanwhile, it fits with the corrected labels $y_{new}$ and the test accuracy on clean labels does not drop. This shows that the model and the label correction are improving in a harmonic fashion and do not collapse. On the right, we show the curves of the Standard method. Without label correction, the model overfits with noisy labels and the performance on test data degrades catastrophically.

## 5. Conclusion

We prove theoretical guarantees for data-re-calibrating methods for noisy labels. Based on the result, we propose a label correction algorithm to combat label noise. Our method can produce models robust to different noise patterns. Experiments on various datasets show that our method outperforms many recently proposed methods.

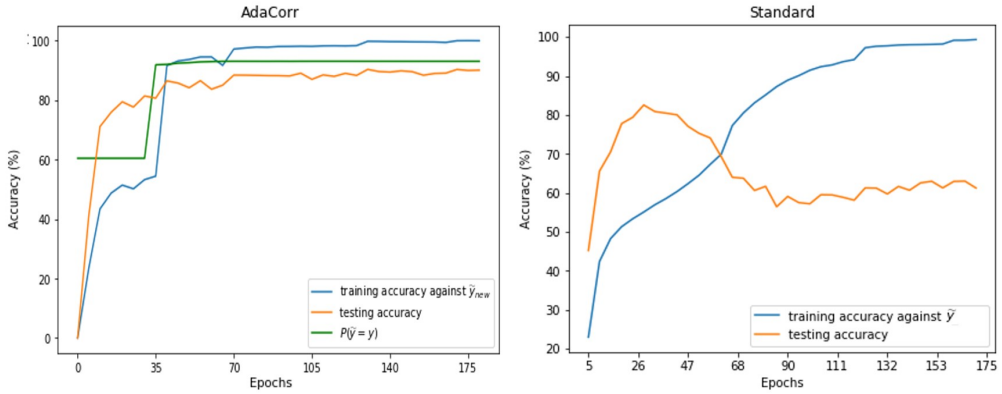



*Figure 3.* Convergence curves for CIFAR10 with 40% uniform noise. Left: AdaCorr - training accuracy evaluated against the corrected label ($y_{new}$) (cyan), testing accuracy against clean label (orange), and the proportion of correct label (green). Right: Standard - training accuracy against noisy label ($\widetilde{y}$) and testing accuracy against clean label.

*Table 2.* The classification accuracy of different methods.

| Data Set | Method | Noise Level of Uniform Flipping | | | | Noise Level of Pair Flipping | | |
|---|---|---|---|---|---|---|---|---|
| | | 0.2 | 0.4 | 0.6 | 0.8 | 0.2 | 0.3 | 0.4 |
| MNIST | Standard | 99.0 ± 0.2 | 98.7 ± 0.4 | 98.1 ± 0.3 | 91.3 ± 0.9 | 99.3 ± 0.1 | 99.2 ± 0.1 | 98.8 ± 0.1 |
| | Forgetting | 99.0 ± 0.1 | 98.8 ± 0.1 | 97.7 ± 0.2 | 62.6 ± 8.9 | 99.3 ± 0.1 | 96.5 ± 2.0 | 89.7 ± 1.9 |
| | Forward | 99.1 ± 0.1 | 98.7 ± 0.2 | 98.0 ± 0.4 | 89.6 ± 4.8 | 99.4 ± 0.0 | 99.2 ± 0.2 | 96.5 ± 4.4 |
| | Decouple | 99.3 ± 0.1 | 99.0 ± 0.1 | 98.5 ± 0.2 | 94.6 ± 0.2 | 99.4 ± 0.0 | 99.3 ± 0.1 | 99.1 ± 0.2 |
| | MentorNet | 99.2 ± 0.2 | 98.7 ± 0.1 | 98.1 ± 0.1 | 87.5 ± 5.2 | 98.6 ± 0.4 | 99.1 ± 0.1 | 98.9 ± 0.1 |
| | Coteach | 99.1 ± 0.2 | 98.7 ± 0.3 | 98.2 ± 0.3 | 95.7 ± 0.7 | 99.1 ± 0.1 | 99.0 ± 0.2 | 98.9 ± 0.2 |
| | Abstention | 94.0 ± 0.3 | 76.8 ± 0.3 | 49.6 ± 0.1 | 21.2 ± 0.5 | 94.3 ± 0.3 | 88.5 ± 0.3 | 81.4 ± 0.2 |
| | AdaCorr | 99.5 ± 0.0 | 99.4 ± 0.0 | 99.1 ± 0.0 | 97.7 ± 0.2 | 99.5 ± 0.0 | 99.6 ± 0.0 | 99.4 ± 0.0 |
| CIFAR10 | Standard | 87.5 ± 0.2 | 83.1 ± 0.4 | 76.4 ± 0.4 | 47.6 ± 2.0 | 88.8 ± 0.2 | 88.4 ± 0.3 | 84.5 ± 0.3 |
| | Forgetting | 87.1 ±0.2 | 83.4 ± 0.2 | 76.5 ± 0.7 | 33.0 ± 1.6 | 89.6 ± 0.1 | 83.7 ± 0.1 | 86.4 ± 0.5 |
| | Forward | 87.4 ± 0.8 | 83.1 ± 0.8 | 74.7 ± 1.7 | 38.3 ± 3.0 | 89.0 ± 0.5 | 87.4 ± 1.1 | 84.7 ± 0.5 |
| | Decouple | 87.6 ± 0.4 | 84.2 ± 0.5 | 77.6 ± 0.1 | 48.5 ± 0.9 | 90.6 ± 0.3 | 89.1 ± 0.3 | 86.3 ± 0.5 |
| | MentorNet | 90.3 ± 0.3 | 83.2 ± 0.5 | 75.5 ± 0.7 | 34.1 ± 2.5 | 90.4 ± 0.2 | 88.9 ± 0.1 | 83.3 ± 1.0 |
| | Coteach | 90.1 ± 0.4 | 87.3 ± 0.5 | 80.9 ± 0.5 | 25.0 ± 3.6 | 91.8 ± 0.1 | 89.9 ± 0.2 | 80.1 ± 0.7 |
| | Abstention | 85.3 ± 0.4 | 82.0 ± 0.7 | 68.8 ± 0.4 | 33.8 ± 7.7 | 88.5 ± 0.0 | 83.1 ± 0.5 | 77.4 ± 0.4 |
| | AdaCorr | **91.0 ± 0.3** | **88.7 ± 0.5** | **81.2 ± 0.4** | **49.2 ± 2.4** | **92.2 ± 0.1** | **91.3 ± 0.3** | **89.2 ± 0.4** |
| CIFAR100 | Standard | 58.9 ± 0.8 | 52.1 ± 1.0 | 42.1 ± 0.7 | 20.8 ± 1.0 | 59.5 ± 0.4 | 52.9 ± 0.6 | 44.7 ± 1.3 |
| | Forgetting | 59.3 ± 0.8 | 53.0 ± 0.2 | 40.9 ± 0.5 | 7.7 ± 1.1 | 61.4 ± 0.9 | 54.6 ± 0.6 | 37.7 ± 4.6 |
| | Forward | 58.4 ± 0.5 | 52.2 ± 0.3 | 41.1 ± 0.5 | 20.6 ± 0.6 | 58.3 ± 0.7 | 53.2 ± 0.6 | 44.4 ± 2.8 |
| | Decouple | 59.0 ± 0.7 | 52.2 ± 0.7 | 40.2 ± 0.4 | 18.5 ± 0.8 | 60.8 ± 0.7 | 56.1 ± 0.7 | 48.4 ± 1.0 |
| | MentorNet | 63.6 ± 0.5 | 51.4 ± 1.4 | 38.7 ± 0.8 | 17.4 ± 0.9 | 64.7 ± 0.2 | 57.4 ± 0.8 | 47.4 ± 1.7 |
| | Coteach | 66.1 ± 0.5 | 60.0 ± 0.6 | **48.3 ± 0.1** | 16.1 ± 1.1 | 63.4 ± 0.9 | 57.6 ± 0.3 | 49.2 ± 0.3 |
| | Abstention | **75.1± 5.4** | 60.0 ± 0.8 | 51.1± 0.8 | 10.3 ± 0.5 | 65.4 ± 0.5 | 56.8 ± 0.5 | 47.3 ± 0.3 |
| | AdaCorr | 67.8 ± 0.1 | **60.2 ± 0.8** | 46.5 ± 1.2 | **24.6 ± 1.1** | **68.3 ± 0.2** | **61.1 ± 0.5** | **49.8 ± 0.7** |
| ModelNet40 | Standard | 79.1 ± 2.6 | 75.3 ± 3.3 | 70.0 ± 3.0 | 57.9 ± 2.3 | 84.4 ± 1.2 | 82.3 ± 1.3 | 78.9 ± 0.7 |
| | Forgetting | 80.1 ± 1.8 | 73.9 ± 0.6 | 69.0 ± 0.7 | 26.2 ± 4.8 | 83.3 ± 1.1 | 62.0 ± 3.0 | 59.5 ± 2.9 |
| | Forward | 52.3 ± 5.1 | 49.4 ± 6.8 | 43.5 ± 5.2 | 28.2 ± 5.5 | 48.1 ± 6.8 | 48.0 ± 3.7 | 49.1 ± 4.4 |
| | Decouple | 82.5 ± 2.2 | 80.7 ± 0.7 | 72.9 ± 1.0 | 55.4 ± 2.7 | 85.7 ± 1.4 | 84.3 ± 1.0 | 80.5 ± 2.4 |
| | MentorNet | 86.5 ± 0.5 | 75.4 ± 1.8 | 70.9 ± 1.9 | 52.7 ± 3.1 | 83.7 ± 1.8 | 81.0 ± 1.5 | 79.3 ± 2.1 |
| | Coteach | 85.6 ± 0.9 | 84.2 ± 0.8 | **81.8 ± 1.1** | 68.9 ± 2.8 | 85.7 ± 0.8 | 79.1 ± 3.0 | 69.1 ± 2.4 |
| | Abstention | 78.1 ± 0.6 | 65.6 ± 0.5 | 45.6 ± 1.5 | 23.5 ± 0.5 | 82.3 ± 0.5 | 80.4 ± 0.6 | 65.6 ± 0.5 |
| | AdaCorr | **86.9 ± 0.3** | **85.1 ± 0.6** | 78.6 ± 1.4 | **72.1 ± 1.1** | **87.6 ± 0.4** | **84.6 ± 0.5** | **83.7 ± 0.5** |

## Acknowledgements

Mayank Goswami is supported by National Science Foundation grants CRII-1755791 and CCF-1910873. The research of Songzhu Zheng and Chao Chen is partially supported by NSF IIS-1855759, CCF-1855760 and IIS-1909038. The research of Pengxiang Wu and Dimitris Metaxas is partially supported by NSF CCF-1733843. We thank anonymous referees for constructive comments and suggestions.

## References

Andreas, V., Neil, A., Gal, C., Ivan, K., Abhinav, G., and Serge J., B. Learning from noisy large-scale datasets with minimal supervision. In *CVPR*, pp. 6575–6583, 2017.

Belkin, M., Hsu, D. J., and Mitra, P. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. In *NeurIPS*, pp. 2300–2311, 2018.

Brodley, C. E. and Friedl, M. A. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11:131–167, 1999.

Chaudhuri, K. and Dasgupta, S. Rates of convergence for nearest neighbor classification. In *NeurIPS*, pp. 3437–3445, 2014.

Chen, D.-R. and Sun, T. Consistency of multiclass empirical risk minimization methods based on convex loss. *Journal of Machine Learning Research*, 7(11):2435–2447, 2006.

Cheng, J., Liu, T., Ramamohanarao, K., and Tao, D. Learning with bounded instance-and label-dependent label noise. In *ICML*, 2020.

Crammer, K. and Lee, D. D. Learning via gaussian herding. In *NeurIPS*, pp. 451–459. 2010.

Crammer, K., Kulesza, A., and Dredze, M. Adaptive regularization of weight vectors. In *NeurIPS*, pp. 414–422, 2009.

Dan, H., Kimin, L., and Mantas, M. Using pre-training can improve model robustness and uncertainty. In *ICML*, pp. 2712–2721, 2019.

Devansh, A., Stanislaw, K. J., Nicolas, B., David, K., Emmanuel, B., Maxinder, S. K., Tegan, M., Asja, F., Aaron, C. C., Yoshua, B., and Simon, L. A closer look at memorization in deep networks. In *ICML*, pp. 233–242, 2017.

Frénay, B. and Verleysen, M. Classification in the presence of label noise: A survey. *Neural Networks and Learning Systems, IEEE Transactions on*, 25(5):845–869, 2014.

Gao, W., Yang, B.-B., and Zhou, Z.-H. On the resistance of nearest neighbor to random noisy labels. *arXiv*, pp. arXiv–1607, 2016.

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

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I. W., and Sugiyama, M. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, pp. 8536–8546, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *ECCV*, pp. 630–645, 2016.

Jacob, S., Pang Wei, K., and Percy S., L. Certified defenses for data poisoning attacks. In *NeurIPS*, pp. 3520–3532, 2017.

Jiang, L., Zhou, Z., Leung, T., Li, J., and Li, F.-F. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, pp. 2304–2313, 2018.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=726791.

Lee, K., Yun, S., Lee, K., Lee, H., Li, B., and Shin, J. Robust inference via generative classifiers for handling noisy labels. In *ICML*, 2019.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. On the variance of the adaptive learning rate and beyond. In *ICLR*, 2019.

Malach, E. and Shalev-Shwartz, S. Decoupling "when to update" from "how to update". In *NeurIPS*, pp. 960–970, 2017.

Nagarajan, N., Ambuj, T., Inderjit S., D., and Pradeep, R. Learning with noisy labels. In *NeurIPS*, 2013.

Patrini, G., Rozza, A., Menon, A., Nock, R., and Qu, L. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, pp. 2233–2241, 2017.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pp. 652–660, 2017.

Qiao, X., Duan, J., and Cheng, G. Rates of convergence for large-scale nearest neighbor classification. In *NeurIPS*, pp. 10768–10779, 2019.

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. Training deep neural networks on noisy labels with bootstrapping. *arXiv*, pp. arXiv–1412, 2014.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. Learning to reweight examples for robust deep learning. In *ICML*, pp. 4334–4343, 2018.

Shen, Y. and Sanghavi, S. Learning with bad training data via iterative trimmed loss minimization. In *ICML*, pp. 5739–5748, 2019.

Smyth, P., Fayyad, U., Burl, M., Perona, P., and Baldi, P. Inferring ground truth from subjective labelling of venus images. In *NeurIPS*, pp. 1085–1092, 1994.

Tanaka, D., Ikami, D., Yamasaki, T., and Aizawa, K. Joint optimization framework for learning with noisy labels. In *CVPR*, pp. 5552–5560, 2018.

Thulasidasan, S., Bhattacharya, T., Bilmes, J., Chennupati, G., and Mohd-Yusof, J. Combating label noise in deep learning using abstention. In *ICML*, pp. 6234–6243, 2019.

Tsybakov, A. B. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135–166, 2004.

Van Rooyen, B., Menon, A., and Williamson, R. C. Learning with symmetric label noise: The importance of being unhinged. In *NeurIPS*, pp. 10–18, 2015.

Volodymyr, M. and Geoffrey E., H. Learning to label aerial images from noisy data. In *ICML*, pp. 567–574, 2012.

Wang, J. and Chaudhuri. Analyzing the robustness of nearest neighbors to adversarial examples. In *ICML*, pp. 5133–5142, 2018.

Wang, Y., Liu, W., Ma, X., Bailey, J., Zha, H., Song, L., and Xia, S.-T. Iterative learning with open-set noisy labels. In *CVPR*, pp. 8688–8696, 2018.

Wu, X., He, R., Sun, Z., and Tan, T. A light CNN for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security*, 13:2884–2896, 2018.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. 3d shapenets: A deep representation for volumetric shape modeling. In *CVPR*, pp. 1912–1920, 2015.

Xiao, T., Xia, T., Yang, Y., Huang, C., and Wang, X. Learning from massive noisy labeled data for image classification. In *CVPR*, pp. 2691–2699, 2015.

Yan, Y., Rosales, R., Fung, G., Subramanian, R., and Jennifer, D. Learning from multiple annotators with varying expertise. *Machine learning*, 95(3):291–327, 2014.

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

Zhang, Z. and Sabuncu, M. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*, pp. 8778–8788, 2018.