Learning from Noisy Labels with No Change to the Training Process

Mingyuan Zhang [1] **Jane Lee** [2] **Shivani Agarwal** [1]

Abstract

There has been much interest in recent years in developing learning algorithms that can learn accurate classifiers from data with noisy labels. A widely-studied noise model is that of *class-conditional noise* (CCN), wherein a label y is flipped to a label $\widetilde{y}$ with some associated noise probability that depends on both y and $\widetilde{y}$. In the multiclass setting, all previously proposed algorithms under the CCN model involve changing the training process, by introducing a 'noise-correction' to the surrogate loss to be minimized over the noisy training examples. In this paper, we show that this is really unnecessary: one can simply perform class probability estimation (CPE) on the noisy examples, e.g. using a standard (multiclass) logistic regression algorithm, and then apply noise-correction only in the final prediction step. This means that the training algorithm itself does not need any change, and one can simply use standard off-the-shelf implementations with no modification to the code for training. Our approach can handle general multiclass loss matrices, including the usual 0-1 loss but also other losses such as those used for ordinal regression problems. We also provide a quantitative regret transfer bound, which bounds the target regret on the true distribution in terms of the CPE regret on the noisy distribution; in doing so, we extend the notion of strong properness introduced for binary losses by Agarwal (2014) to the multiclass case. Our bound suggests that the sample complexity of learning under CCN increases as the noise matrix approaches singularity. We also provide fixes and potential improvements for noise estimation methods that involve computing anchor points. Our experiments confirm our theoretical findings.

[1]Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA [2]Twitter, San Francisco, CA, USA (Work done while at the University of Pennsylvania). Correspondence to: Mingyuan Zhang, Shivani Agarwal <{myz,ashivani}@seas.upenn.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

1. Introduction

In many applications of machine learning, one receives noisy labels during training. This can happen for a variety of reasons, including human labeling errors, sensor measurement errors, distributed label collection via crowdsourcing, automatic label collection via internet crawling, and many others. Consequently, there has been much interest in recent years in developing learning algorithms that can learn accurate classifiers from data with noisy labels (Frénay & Verleysen, 2014; Song et al., 2020).

We focus here on the setting of *label-dependent noise*, where the (random) noise in a label depends on the label but not on the instance (the more general setting of *label- and instance-dependent noise* is also of interest (Menon et al., 2018; Cheng et al., 2020), but we do not focus on that here). An early example of label-dependent noise for binary classification that has been widely studied in the PAC learning literature is the *random classification noise* (RCN) model, in which a binary label y is flipped to the opposite label with a fixed probability $\gamma \in [0, \frac{1}{2})$ (Angluin & Laird, 1987; Bylander, 1994; Aslam & Decatur, 1996; Kearns, 1998; Blum & Mitchell, 1998; Cesa-Bianchi et al., 1999). More recently, Natarajan et al. (2013) generalized the RCN model and proposed the *class-conditional random label noise* (CCN) model for binary classification, in which flip probabilities for positive and negative labels can be different. This was then extended to the more general multiclass case, wherein a label y is flipped to a label $\widetilde{y}$ with some noise probability that depends on y and $\widetilde{y}$ (van Rooyen & Williamson, 2017; Patrini et al., 2017; Ghosh et al., 2017; Wang et al., 2018).

The primary challenge in learning from noisy labels is to design algorithms which, despite being given data with noisy labels as input, can learn accurate classifiers for the true, *clean* distribution. In particular, it is desirable to design algorithms which, when trained using a sufficiently rich function class, are statistically consistent for the clean distribution (i.e. that converge to a Bayes optimal classifier for the clean distribution). For the general multiclass CCN model, two such algorithms have been proposed: the *unbiased estimator* method of van Rooyen & Williamson (2017) (which builds on a method of Natarajan et al. (2013) for binary labels), and the *forward* method of Patrini et al. (2017) (the 'backward' method of Patrini et al. (2017) is the same as

the unbiased estimators method). Both algorithms make use of the framework of surrogate loss minimization, and both require modifying the surrogate loss to correct for the noise. In practice, this means modifying the training algorithm.

In this paper, we take a first-principles approach, and show that, for the general multiclass CCN model, one can design statistically consistent learning algorithms *without* modifying the training process. In particular, by examining the form of the Bayes optimal classifier for any target (cost-sensitive) multiclass loss, and the relation between the noisy and clean distributions over labels, we show that it suffices to simply implement a standard class probability estimation (CPE) algorithm (such as multiclass logistic regression) on the given noisy training data, and then apply a noise-corrected plug-in step at prediction time. For practitioners lacking expertise to modify the optimization process, or when retraining is a bottleneck, the post-processing step at prediction time can be easier to implement and use.

To establish consistency of our method (when trained with a sufficiently rich function class), we derive a quantitative regret transfer bound which shows that the target regret on the true, clean distribution can be upper bounded by the CPE regret on the noisy distribution. We also extend the notion of strong properness, defined for binary losses by Agarwal (2014), to multiclass surrogate losses; for CPE learners that minimize such surrogate losses (including for example the multiclass logistic/cross-entropy loss), we provide a regret bound in terms of the surrogate regret on the noisy distribution. Our bound suggests that as the noise matrix becomes closer to being singular, the sample size needed to achieve a given target performance level becomes larger.

In their basic forms, the methods of both van Rooyen & Williamson (2017) and Patrini et al. (2017), as well as our noise-corrected plug-in method, all assume that the noise flip probabilities are known. In practice, one may need to estimate the noise probabilities from the given noisy data. In recent years, a number of approaches have been proposed for estimating noise flip probabilities; these are generally based on identifying a small number of *anchor points* (instances that belong to a certain class with probability one). In particular, Patrini et al. (2017) proposed a noise estimation method based on anchor points, with the intent to provide an 'end-to-end' noise-estimation-and-learning method. Later, Yao et al. (2020) exploited the divide-and-conquer paradigm to propose another noise estimation method, also based on anchor points. However, it turns out that both methods do not always work correctly; we identify an error in their methods (specifically, the error is in the method for computing anchor points), and provide conditions on the noise under which the methods work or fail. We also propose an iterative noise estimation heuristic that aims to partly correct the error; while the heuristic is not guaranteed to converge or recover the correct noise probabilities, it works well in our experiments, sometimes outperforming the methods of Patrini et al. (2017) and Yao et al. (2020). Moreover, all three noise estimation methods require a CPE model to be learned from the noisy data, which in our case comes for free, with no further training required; thus our method also provides a more efficient 'end-to-end' solution.

Our experiments confirm that our noise-corrected plug-in method performs comparably to previous methods, while requiring no change to the training process.

Relationship with previous work in the binary case. As noted above, the works on learning from noisy labels in multiclass classification that are most closely related to ours are those of van Rooyen & Williamson (2017) and Patrini et al. (2017). In the special case of binary classification, two works are most directly relevant: those of Natarajan et al. (2013) and Menon et al. (2015). Natarajan et al. (2013) studied the CCN model for binary classification, and expressed the Bayes optimal classifier for the *noisy* distribution as a plug-in rule involving the *clean* class probability function (Lemma 7), and then used this to reduce the CCN learning problem to a cost-sensitive classification problem on the noisy data using classification-calibrated surrogate losses. In contrast, we express the Bayes optimal classifier for the *clean* distribution as a plug-in rule involving the *noisy* class probabilities, which can be estimated directly from the noisy data (we use strongly proper composite surrogate losses for this estimation). Menon et al. (2015) studied the more general *mutually contaminated distributions* (MCD) noise model for binary classification, and while they focused mostly on the balanced error (BER) and area under the ROC curve (AUC) metrics, they also used strongly proper composite (binary) surrogate losses, and applied their analysis to derive a regret transfer bound for the 0-1 error as well (Proposition 7). When specialized to the CCN model, their bound for binary classification with 0-1 loss can be viewed as a special case of our bound in Theorem 4 (our bound holds for multiclass classification with general losses).

Organization. After preliminaries in Section 2, we describe our noise-corrected plug-in method in Section 3. Section 4 gives regret transfer bounds; Section 5 discusses noise estimation. Section 6 summarizes our experiments. All proofs can be found in the supplementary material.

Notation. For an integer n, we denote by $[n]$ the set of integers $\{1, \ldots, n\}$, and by Δ_n the probability simplex $\{\mathbf{p} \in \mathbb{R}^n_+ : \sum_{y=1}^n p_y = 1\}$. For a vector $\mathbf{a}$, we denote by $\|\mathbf{a}\|_2$ the L_2 norm of $\mathbf{a}$. For a matrix $\mathbf{A}$, we denote by $\|\mathbf{A}\|_F$ the Frobenius norm of $\mathbf{A}$, by $\|\mathbf{A}\|_2$ the induced 2-norm of $\mathbf{A}$ (largest singular value of $\mathbf{A}$), and by $\mathbf{a}_y$ the y-th column vector of $\mathbf{A}$. We use $\mathbf{e}_y$ to denote a standard basis vector with y-th element 1.

2. Preliminaries

The problem of (multiclass) learning from noisy labels can be described as follows. There is an instance space $\mathcal{X}$, and a set of n class labels $\mathcal{Y}$, which we will take without loss of generality to be $\mathcal{Y} = [n]$. There is a (unknown) joint probability distribution D on $\mathcal{X} \times \mathcal{Y}$ from which labeled examples (X, Y) are drawn. In the standard (non-noisy) supervised learning setting, the learner would be given training examples drawn directly from D. When learning from noisy labels, however, the learner does not get clean labels Y; instead, the learner sees noisy examples $(X, \widetilde{Y})$, where $\widetilde{Y}$ denotes a noisy version of Y. In particular, the learner receives a noisy training sample $\widetilde{S} = ((x_1, \widetilde{y}_1), \ldots, (x_m, \widetilde{y}_m)) \in (\mathcal{X} \times \mathcal{Y})^m$, and the goal is to learn a classifier $h : \mathcal{X} \to \mathcal{Y}$ that performs well with respect to the clean distribution D.

We consider here the *class-conditional* random label noise (CCN) model (Natarajan et al., 2013; van Rooyen & Williamson, 2017), wherein a label y is randomly flipped to a label $\widetilde{y}$ with some probability $\gamma_{y,\widetilde{y}}$ that depends on y and $\widetilde{y}$. In particular, the CCN model is characterized by a row-stochastic *noise matrix* $\mathbf{C} \in [0,1]^{n \times n}$ with entries $\gamma_{y,\widetilde{y}}$, such that for each $y, \widetilde{y} \in [n]$,

$$\mathbf{P}(\widetilde{Y} = \widetilde{y} \,|\, Y = y) = \gamma_{y,\widetilde{y}} \,.$$

The noisy training examples seen by the learner can therefore be viewed as being drawn IID from a 'noisy' distribution $\widetilde{D}$ on $\mathcal{X} \times \mathcal{Y}$, wherein an example (X, Y) is first drawn randomly according to D, and then noise is injected according to the noise matrix $\mathbf{C}$ to generate $(X, \widetilde{Y})$.

Thus, given a noisy training sample $\widetilde{S}$ drawn according to the noisy distribution $\widetilde{D}$ as above, the goal of the learner is to learn a classifier $h : \mathcal{X} \to \mathcal{Y}$ that performs well under the clean distribution D. To measure performance, we consider a general multiclass loss matrix $\mathbf{L} \in \mathbb{R}_+^{n \times n}$, with entries $\ell_{y,\widehat{y}}$ indicating the loss incurred on predicting $\widehat{y}$ when the true label is y (the 0-1 loss $\mathbf{L}^{\text{0-1}}$ with $\ell^{\text{0-1}}_{y,\widehat{y}} = \mathbf{1}(\widehat{y} \neq y)$ is a special case). The performance of the classifier h is then measured by the $\mathbf{L}$-generalization error or $\mathbf{L}$-risk under D:

$$\text{er}^{\mathbf{L}}_D[h] = \mathbf{E}_{(X,Y) \sim D}\left[\ell_{Y,h(X)}\right] .$$

3. Noise-Corrected Plug-in Method

The approach we describe is conceptually very simple. We will denote by $\boldsymbol{\eta}, \widetilde{\boldsymbol{\eta}} : \mathcal{X} \to \Delta_n$ the (vector) class probability functions under the clean distribution D associated with clean labeled examples and the noisy distribution $\widetilde{D}$ associated with noisy examples, respectively, with components given by

$$\begin{aligned} \eta_y(x) &= \mathbf{P}(Y = y \,|\, X = x) \\ \widetilde{\eta}_y(x) &= \mathbf{P}(\widetilde{Y} = y \,|\, X = x) \end{aligned}$$

for each $y \in [n]$. It is easy to see that

$$\begin{aligned} \widetilde{\eta}_y(x) &= \sum_{y' \in [n]} \mathbf{P}(\widetilde{Y} = y \,|\, Y = y') \cdot \mathbf{P}(Y = y' \,|\, X = x) \\ &= \sum_{y' \in [n]} \gamma_{y',y} \cdot \eta_{y'}(x) \\ &= \mathbf{c}_y^\top \boldsymbol{\eta}(x) \,, \end{aligned}$$

which gives

$$\widetilde{\boldsymbol{\eta}}(x) = \mathbf{C}^\top \boldsymbol{\eta}(x) \,.$$

Therefore, provided $\mathbf{C}$ is invertible, we have

$$\boldsymbol{\eta}(x) = (\mathbf{C}^\top)^{-1} \widetilde{\boldsymbol{\eta}}(x) \,. \tag{1}$$

This suggests that once we have an estimate of the *noisy* class probability function $\widetilde{\boldsymbol{\eta}}$, we may be able to 'de-noise' it to construct an estimate of the clean class probability function $\boldsymbol{\eta}$. This idea in its basic form can be problematic, since $\mathbf{C}^{-1}$ is not necessarily a stochastic matrix; in particular, $\mathbf{C}^\top$ generally maps probability vectors $\boldsymbol{\eta}(x)$ in the probability simplex Δ_n to noisy probability vectors $\widetilde{\boldsymbol{\eta}}(x)$ in a limited *subset* of the simplex Δ_n, and in general, an estimate of $\widetilde{\boldsymbol{\eta}}(x)$ could fall outside that subset, so that multiplying the estimate by $(\mathbf{C}^\top)^{-1}$ could then lead to an invalid 'estimate' of $\boldsymbol{\eta}(x)$ that falls outside Δ_n. Nevertheless, we get around this issue by never really needing to construct a fully valid estimate of $\boldsymbol{\eta}(x)$; instead, we simply use the above relation to derive a noise-corrected plug-in classifier that operates directly on estimates of the *noisy* class probabilities $\widetilde{\boldsymbol{\eta}}(x)$. Our regret transfer bounds in Section 4 will establish that this indeed leads to a correct learning approach.

We start by explaining our approach in the context of the multiclass 0-1 loss, and then describe the extension to general multiclass losses.

Multiclass 0-1 loss. As is well known, the Bayes optimal classifier for the multiclass 0-1 loss is given by

$$h^{\text{0-1},*}_D(x) = \operatorname*{argmax}_{y \in [n]} \eta_y(x) \,.$$

By Eq. (1), we can re-write this in terms of the *noisy* class probability function $\widetilde{\boldsymbol{\eta}}$ as follows:

$$\begin{aligned} h^{\text{0-1},*}_D(x) &= \operatorname*{argmax}_{y \in [n]} \left((\mathbf{C}^\top)^{-1} \widetilde{\boldsymbol{\eta}}(x)\right)_y \\ &=: \text{plugin}^{\text{0-1}}_{\mathbf{C}}\left(\widetilde{\boldsymbol{\eta}}(x)\right) . \end{aligned}$$

Notably, this means that during training, we can simply construct a multiclass CPE model $\widehat{\widetilde{\boldsymbol{\eta}}} : \mathcal{X} \to \Delta_n$ for the *noisy* class probability function $\widetilde{\boldsymbol{\eta}}$, by running any standard multiclass CPE method (such as standard multiclass logistic regression) on the given *noisy* training examples, and then construct a noise-corrected classifier $\widehat{h} : \mathcal{X} \to \mathcal{Y}$ by applying the above noise-corrected plug-in step during prediction:

$$\widehat{h}(x) = \text{plugin}^{\text{0-1}}_{\mathbf{C}}\left(\widehat{\widetilde{\boldsymbol{\eta}}}(x)\right) .$$

Multiclass cost-sensitive losses. More generally, consider any multiclass loss matrix $\mathbf{L} \in \mathbb{R}_+^{n \times n}$. The Bayes optimal

Algorithm 1 Noise-Corrected Plug-in Algorithm

Inputs:
(1) Noisy training sample, $\widetilde{S} = ((x_1, \widetilde{y}_1), \ldots, (x_m, \widetilde{y}_m)) \in (\mathcal{X} \times \mathcal{Y})^m$
(2) Target loss matrix $\mathbf{L} \in \mathbb{R}_+^{n \times n}$
(3) (If known) Noise matrix $\mathbf{C} \in [0,1]^{n \times n}$
Run a standard CPE learner on $\widetilde{S}$:
$\widehat{\widetilde{\boldsymbol{\eta}}} = \text{CPE-Learner}(\widetilde{S})$
If $\mathbf{C}$ unknown: Construct estimate $\widehat{\mathbf{C}}$ (see Section 5)
Output:
If $\mathbf{C}$ known: $\widehat{h} = \text{plugin}_{\mathbf{C}}^{\mathbf{L}} \circ \widehat{\widetilde{\boldsymbol{\eta}}}$
If $\mathbf{C}$ unknown: $\widehat{h} = \text{plugin}_{\widehat{\mathbf{C}}}^{\mathbf{L}} \circ \widehat{\widetilde{\boldsymbol{\eta}}}$

classifier for $\mathbf{L}$ (which for any instance x, chooses a prediction that minimizes the expected loss under $\mathbf{L}$) is given by

$$h_D^{\mathbf{L},*}(x) \;=\; \operatorname*{argmin}_{y \in [n]} \boldsymbol{\eta}(x)^\top \boldsymbol{\ell}_y \,.$$

As for the 0-1 loss, by Eq. (1), we can re-write this in terms of the *noisy* class probability function $\widetilde{\boldsymbol{\eta}}$ as follows:

$$\begin{aligned} h_D^{\mathbf{L},*}(x) &= \operatorname*{argmin}_{y \in [n]} \widetilde{\boldsymbol{\eta}}(x)^\top \mathbf{C}^{-1} \boldsymbol{\ell}_y \\ &= \operatorname*{argmin}_{y \in [n]} \widetilde{\boldsymbol{\eta}}(x)^\top \left(\mathbf{C}^{-1}\mathbf{L}\right)_y \\ &=: \text{plugin}_{\mathbf{C}}^{\mathbf{L}}\left(\widetilde{\boldsymbol{\eta}}(x)\right). \end{aligned}$$

Again, this means that during training, we can use a standard multiclass CPE learner on the noisy examples to construct a CPE model $\widehat{\widetilde{\boldsymbol{\eta}}} : \mathcal{X} \to \Delta_n$ for the *noisy* class probability function $\widetilde{\boldsymbol{\eta}}$, and then construct a noise-corrected classifier $\widehat{h} : \mathcal{X} \to \mathcal{Y}$ by applying the above noise-corrected plug-in step during prediction:

$$\widehat{h}(x) = \text{plugin}_{\mathbf{C}}^{\mathbf{L}}\left(\widehat{\widetilde{\boldsymbol{\eta}}}(x)\right).$$

Note that one can pre-compute $\mathbf{C}^{-1}\mathbf{L}$, and so at prediction time, in order to implement $\text{plugin}_{\mathbf{C}}^{\mathbf{L}}(\widehat{\widetilde{\boldsymbol{\eta}}}(x))$, one needs to compute n inner products (of the column vectors of $\mathbf{C}^{-1}\mathbf{L}$ with $\widehat{\widetilde{\boldsymbol{\eta}}}(x)$), for a total computational cost of $O(n^2)$.[1,2]

Our final algorithm is shown in Algorithm 1. An example of a CPE learner that minimizes a (strongly) proper composite multiclass surrogate loss is provided in Section 4.2. In settings where the noise matrix $\mathbf{C}$ is not known, one may need to estimate $\mathbf{C}$ from the noisy training sample itself; this is discussed in Section 5.

[1] Also note that if one has an implementation of the standard plug-in step $\text{plugin}^{\mathbf{L}}(\cdot)$ (without noise correction) for general (cost-sensitive) loss matrices $\mathbf{L}$, one can simply use that implementation with loss $\widetilde{\mathbf{L}} = \mathbf{C}^{-1}\mathbf{L}$ (since $\text{plugin}_{\mathbf{C}}^{\mathbf{L}}\left(\widetilde{\boldsymbol{\eta}}(x)\right) = \text{plugin}^{\widetilde{\mathbf{L}}}\left(\widetilde{\boldsymbol{\eta}}(x)\right)$).

[2] This also applies to the 0-1 loss: one can simply pre-compute $\mathbf{C}^{-1}\mathbf{L}^{0\text{-}1}$, and then at prediction time, compute n inner products (of the column vectors of $\mathbf{C}^{-1}\mathbf{L}^{0\text{-}1}$ with $\widehat{\widetilde{\boldsymbol{\eta}}}(x)$) for a cost of $O(n^2)$ (and predict according to $\operatorname{argmin}_{y \in [n]} \widehat{\widetilde{\boldsymbol{\eta}}}(x)^\top (\mathbf{C}^{-1}\mathbf{L}^{0\text{-}1})_y$).

4. Regret Transfer Bounds and Consistency

In this section, we provide quantitative regret transfer bounds for our noise-corrected plug-in algorithm; these bounds also establish that if the noisy CPE method used in training is consistent (i.e., converges to the correct noisy class probabilities), then our approach is consistent for the target learning problem. We derive our results for the multiclass case with a general loss matrix $\mathbf{L}$; they can be specialized to the binary and/or 0-1 case as needed. In particular, define the $\mathbf{L}$-*regret* (or the *excess* $\mathbf{L}$-*risk*) of a classifier $h : \mathcal{X} \to \mathcal{Y}$ under the clean distribution D as follows:

$$\text{regret}_D^{\mathbf{L}}[h] \;=\; \text{er}_D^{\mathbf{L}}[h] - \inf_{h' : \mathcal{X} \to \mathcal{Y}} \text{er}_D^{\mathbf{L}}[h'] \,. \qquad (2)$$

Our goal is to upper bound this $\mathbf{L}$-regret for our learned classifier $\widehat{h}$; if this regret converges (in probability, over the random draw of the noisy training sample) to zero as the training sample size increases, then the algorithm is *(Bayes) consistent* for $\mathbf{L}$ under D.

In Section 4.1, we provide a general result upper bounding the target $\mathbf{L}$-regret of our learned classifier $\widehat{h} = \text{plugin}_{\mathbf{C}}^{\mathbf{L}} \circ \widehat{\widetilde{\boldsymbol{\eta}}}$ (on the clean distribution D) in terms of the noisy CPE regret of $\widehat{\widetilde{\boldsymbol{\eta}}}$ (on the noisy distribution $\widetilde{D}$). In Section 4.2, we specialize our result to CPE methods that learn $\widehat{\widetilde{\boldsymbol{\eta}}}$ by minimizing a *strongly proper composite* surrogate loss (extending the notion of strong properness defined for binary losses by Agarwal (2014) to the multiclass case), and apply this result in particular to the multiclass logistic loss, which we show to be 1-strongly proper composite.

4.1. Regret Transfer Bound for General CPE Methods

We have the following result for our noise-corrected plug-in method using any CPE learner:

Theorem 1. *For any noisy CPE model $\widehat{\widetilde{\boldsymbol{\eta}}} : \mathcal{X} \to \Delta_n$ and resulting noise-corrected plug-in classifier $\widehat{h} = \text{plugin}_{\mathbf{C}}^{\mathbf{L}} \circ \widehat{\widetilde{\boldsymbol{\eta}}}$, we have*

$$\begin{aligned} &\text{regret}_D^{\mathbf{L}}[\widehat{h}\,] \\ &\leq\; 2 \max_y \left\|\boldsymbol{\ell}_y\right\|_2 \cdot \left\|\mathbf{C}^{-1}\right\|_2 \cdot \mathbf{E}_X\left[\left\|\widehat{\widetilde{\boldsymbol{\eta}}}(X) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2\right]. \end{aligned}$$

In other words, if the learned noisy CPE model $\widehat{\widetilde{\boldsymbol{\eta}}}$ is close to the correct noisy class probabilities $\widetilde{\boldsymbol{\eta}}$, in the sense that $\mathbf{E}_X\left[\left\|\widehat{\widetilde{\boldsymbol{\eta}}}(X) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2\right]$ is small, then the target $\mathbf{L}$-regret of the noise-corrected plug-in classifier on the clean distribution D, $\text{regret}_D^{\mathbf{L}}[\widehat{h}\,]$, is also small. In particular, if the CPE learner is consistent for the noisy distribution in the sense that $\mathbf{E}_X\left[\left\|\widehat{\widetilde{\boldsymbol{\eta}}}(X) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2\right] \xrightarrow{P} 0$ (as the sample size increases), then the overall noise-corrected plug-in method is (Bayes) $\mathbf{L}$-consistent for the clean distribution D, in the sense that $\text{regret}_D^{\mathbf{L}}[\widehat{h}\,] \xrightarrow{P} 0$.

Note that the above bound depends on the noise matrix $\mathbf{C}$ through the term $\left\|\mathbf{C}^{-1}\right\|_2$. This is the largest singular

value of $\mathbf{C}^{-1}$, or equivalently, the reciprocal of the smallest singular value of $\mathbf{C}$. Thus, as the noise matrix $\mathbf{C}$ approaches singularity, the bound becomes larger. This suggests that as $\mathbf{C}$ becomes closer to being singular, we may need a higher quality class probability approximation on the noisy distribution $\widetilde{D}$ (i.e. larger sample size) to reach the same level of $\mathbf{L}$-regret on the clean distribution D. As we will see in Section 6, our experiments also support this observation.

4.2. Regret Transfer Bound for CPE Methods Minimizing a Strongly Proper Composite Surrogate Loss

In practice, a popular approach for learning CPE models is to minimize a suitable (convex) surrogate loss, such as the multiclass logistic loss (this is also what we use in our experiments). We show that for a suitable class of such surrogate losses, the CPE regret can be further upper bounded in terms of the surrogate loss based regret.

Specifically, let $\psi : [n] \times \mathbb{R}^{n-1} \to \mathbb{R}_+$ be any surrogate loss that acts on $(n-1)$-dimensional 'score vectors' in $\mathbb{R}^{n-1}$, and let $\boldsymbol{\lambda} : \Delta_n \to \mathbb{R}^{n-1}$ be an invertible 'link' function.[3] Then ψ is said to be *strictly proper composite with link function* $\boldsymbol{\lambda}$ if for all $\mathbf{p} \in \Delta_n$ and $\mathbf{u} \in \mathbb{R}^{n-1}$, $\mathbf{u} \neq \boldsymbol{\lambda}(\mathbf{p})$:

$$\mathbf{E}_{Y \sim \mathbf{p}}\big[\psi(Y, \mathbf{u}) - \psi(Y, \boldsymbol{\lambda}(\mathbf{p}))\big] > 0\,.$$

It is well known that minimizing such a strictly proper composite surrogate loss over a suitably rich function class provides consistent class probability estimates (Williamson et al., 2016). For binary surrogates, Agarwal (2014) defined a stronger condition that allows the derivation of quantitative bounds in terms of the surrogate regret. Here we extend this notion to the multiclass case and apply it to obtain bounds for our noisy labels problem.[4]

Definition 2 (Strongly proper composite multiclass losses)**.** *Let $s > 0$. We say a multiclass surrogate loss $\psi : [n] \times \mathbb{R}^{n-1} \to \mathbb{R}_+$ is s-*strongly proper composite with (invertible) link function $\boldsymbol{\lambda} : \Delta_n \to \mathbb{R}^{n-1}$ *if for all $\mathbf{p} \in \Delta_n$ and $\mathbf{u} \in \mathbb{R}^{n-1}$:*

$$\mathbf{E}_{Y \sim \mathbf{p}}\big[\psi(Y, \mathbf{u}) - \psi(Y, \boldsymbol{\lambda}(\mathbf{p}))\big] \;\geq\; \frac{s}{2}\big\|\boldsymbol{\lambda}^{-1}(\mathbf{u}) - \mathbf{p}\big\|_2^2\,.$$

As a concrete example, consider the widely used multiclass logistic surrogate loss:

Example 1 (Multiclass logistic loss and link function)**.** *The multiclass logistic loss $\psi_{\text{mlog}} : [n] \times \mathbb{R}^{n-1} \to \mathbb{R}_+$ is defined as*

$$\psi_{\text{mlog}}(y, \mathbf{u}) = \begin{cases} -\ln\left(\frac{\exp(u_y)}{1+\sum_{i=1}^{n-1}\exp(u_i)}\right) & \text{if } y \in [n-1] \\ \ln\left(1+\sum_{i=1}^{n-1}\exp(u_i)\right) & \text{if } y = n\,. \end{cases}$$

The loss is often used with the invertible link function $\boldsymbol{\lambda}_{\text{mlog}} : \Delta_n \to \mathbb{R}^{n-1}$, which together with its inverse $\boldsymbol{\lambda}_{\text{mlog}}^{-1} : \mathbb{R}^{n-1} \to \Delta_n$ is given by

$$\boldsymbol{\lambda}_{\text{mlog}}(\mathbf{p}) = \begin{pmatrix} \ln(\frac{p_1}{p_n}) \\ \vdots \\ \ln(\frac{p_{n-1}}{p_n}) \end{pmatrix}; \boldsymbol{\lambda}_{\text{mlog}}^{-1}(\mathbf{u}) = \begin{pmatrix} \frac{\exp(u_1)}{1+\sum_{i=1}^{n-1}\exp(u_i)} \\ \vdots \\ \frac{\exp(u_{n-1})}{1+\sum_{i=1}^{n-1}\exp(u_i)} \\ \frac{1}{1+\sum_{i=1}^{n-1}\exp(u_i)} \end{pmatrix}.$$

We note that the multiclass logistic loss above is often implemented in an 'over-parameterized' form, with score vectors in $\mathcal{C} = \mathbb{R}^n$ and the softmax function used for 'inverting' such score vectors to class probabilities (indeed, softmax is a many-to-one mapping).[5] We have the following result showing that ψ_{mlog} is strongly proper composite:

Lemma 3. *The multiclass logistic loss ψ_{mlog} is 1-strongly proper composite with link function $\boldsymbol{\lambda}_{\text{mlog}}$.*

A CPE learner minimizing a strongly proper composite surrogate loss (over noisy training examples) is shown in Algorithm 2. (Instantiating this with the multiclass logistic loss ψ_{mlog} above and the class of linear scoring functions leads to the multiclass linear logistic regression algorithm.)

In what follows, for a surrogate loss $\psi : [n] \times \mathbb{R}^{n-1} \to \mathbb{R}_+$, we will define the *$\psi$-generalization error* of a scoring function $\mathbf{f} : \mathcal{X} \to \mathbb{R}^{n-1}$ under $\widetilde{D}$ as

$$\text{er}_{\widetilde{D}}^{\psi}[\mathbf{f}] \;=\; \mathbf{E}_{(X,Y)\sim\widetilde{D}}\big[\psi(Y, \mathbf{f}(X))\big]\,,$$

and the *ψ-regret* of $\mathbf{f}$ under $\widetilde{D}$ as

$$\text{regret}_{\widetilde{D}}^{\psi}[\mathbf{f}] \;=\; \text{er}_{\widetilde{D}}^{\psi}[\mathbf{f}] - \inf_{\mathbf{f}' : \mathcal{X} \to \mathbb{R}^{n-1}} \text{er}_{\widetilde{D}}^{\psi}[\mathbf{f}']\,.$$

Then we have the following regret transfer bound:

Theorem 4. *Let $s > 0$. Let $\psi : [n] \times \mathbb{R}^{n-1} \to \mathbb{R}_+$ be a s-strongly proper composite surrogate loss with (invertible) link function $\boldsymbol{\lambda} : \Delta_n \to \mathbb{R}^{n-1}$. For any scoring model $\widehat{\widetilde{\mathbf{f}}} : \mathcal{X} \to \mathbb{R}^{n-1}$ being used as a (noisy) CPE model via $\widehat{\widetilde{\boldsymbol{\eta}}}(x) = \boldsymbol{\lambda}^{-1}(\widehat{\widetilde{\mathbf{f}}}(x))$, and resulting noise-corrected plug-in classifier $\widehat{h} = \text{plugin}_{\mathbf{C}}^{\mathbf{L}} \circ (\boldsymbol{\lambda}^{-1} \circ \widehat{\widetilde{\mathbf{f}}})$, we have*

$$\text{regret}_D^{\mathbf{L}}[\widehat{h}\,] \;\leq\; 2 \max_y \big\|\boldsymbol{\ell}_y\big\|_2 \cdot \big\|\mathbf{C}^{-1}\big\|_2 \cdot \sqrt{\frac{2}{s}\,\text{regret}_{\widetilde{D}}^{\psi}[\widehat{\widetilde{\mathbf{f}}}\,]}\,.$$

Thus in particular, if the CPE learner in Algorithm 2 minimizes a strongly proper composite surrogate loss ψ over a universal function class $\mathcal{F}$ (with suitable regularization), thus ensuring that $\text{regret}_{\widetilde{D}}^{\psi}[\widehat{\widetilde{\mathbf{f}}}\,] \xrightarrow{P} 0$, then we have that $\text{regret}_D^{\mathbf{L}}[\widehat{h}\,] \xrightarrow{P} 0$ as desired.

[3] More generally, one can consider surrogate losses $\psi : [n] \times \mathcal{C} \to \mathbb{R}_+$ acting on score vectors in any convex set $\mathcal{C}$ that is in 1-to-1 correspondence with Δ_n, such as $\mathcal{C} = \{\mathbf{u} \in \mathbb{R}^n : \sum_{i=1}^n u_i = 0\}$. It is also common to consider 'over-parameterized' surrogate losses acting on $\mathcal{C} = \mathbb{R}^n$; e.g. see the discussion on the multiclass logistic surrogate loss toward the end of the section.

[4] Strong properness implies strict properness. Most commonly used strictly proper composite losses are also strongly proper composite, but the latter condition allows for stronger quantitative guarantees.

[5] The over-parameterized multiclass logistic loss is also sometimes referred to as the cross-entropy loss.

Algorithm 2 CPE Learner Minimizing a Strongly Proper Composite Surrogate Loss (on Noisy Data)

Input: Noisy training sample, $\widetilde{S} = ((x_1, \widetilde{y}_1), \ldots, (x_m, \widetilde{y}_m)) \in (\mathcal{X} \times \mathcal{Y})^m$
Parameters:
(1) Strongly proper composite loss $\psi : [n] \times \mathbb{R}^{n-1} \to \mathbb{R}_+$ with (invertible) link function $\boldsymbol{\lambda} : \Delta_n \to \mathbb{R}^{n-1}$;
(2) Class $\mathcal{F}$ of functions $\mathbf{f} : \mathcal{X} \to \mathbb{R}^{n-1}$
Compute $\widehat{\widetilde{\mathbf{f}}} \in \operatorname{argmin}_{\mathbf{f} \in \mathcal{F}} \sum_{i=1}^m \psi(\widetilde{y}_i, \mathbf{f}(x_i))$
Output: $\widehat{\widetilde{\boldsymbol{\eta}}} = \boldsymbol{\lambda}^{-1} \circ \widehat{\widetilde{\mathbf{f}}}$

5. Estimating the Noise Matrix $\mathbf{C}$

In settings where the noise matrix $\mathbf{C}$ is not known in advance, one may need to estimate $\mathbf{C}$ from the noisy training examples themselves. Most previous work on estimating the noise matrix assumes the existence of 'anchor points' (definition provided below), and relies on estimating these points accurately.[6] In particular, Menon et al. (2015) provided a method for estimating $\mathbf{C}$ using anchor points in the case of binary labels; Patrini et al. (2017) extended it to the multiclass setting, and later, Yao et al. (2020) proposed another noise estimation method also based on anchor points. Unfortunately, however, we show below that these methods do not work correctly for all noise matrices $\mathbf{C}$. In particular, in Section 5.1, we point out an error in the approach used to compute anchor points in the noise estimation methods of Patrini et al. and Yao et al., and provide sufficient and necessary conditions on $\mathbf{C}$ under which these methods do work correctly. Of course, when $\mathbf{C}$ is unknown, we may not know whether it satisfies these conditions, and so we may not be able to verify whether the estimation is correct. Building on the intuition developed from our analysis, in Section 5.2 we propose an iterative noise estimation heuristic that essentially tries to improve the estimation of anchor points, and that can be applied for any unknown $\mathbf{C}$; while it is not guaranteed to converge or recover a correct estimate, in our experiments, it generally performs as well as, or improves upon, the methods of Patrini et al. and Yao et al. It remains an open question whether general noise matrices $\mathbf{C}$ can be estimated reliably using anchor points.

5.1. Conditions for Correctness of Noise Estimation Methods Based on Anchor Points

The methods of Patrini et al. (2017) and Yao et al. (2020) make the following assumption:

(A) (Anchor points) Under the clean distribution $D = (\mu, \boldsymbol{\eta})$, for every $y \in \mathcal{Y}$, there is a 'perfect' example $\bar{x}^y \in \mathcal{X}$ of class y (called an *anchor point* of class y) with marginal $\mu(\bar{x}^y) > 0$ and $\boldsymbol{\eta}(\bar{x}^y) = \mathbf{e}_y$.

[6] There is also some recent work that aims to estimate $\mathbf{C}$ without identifying anchor points (Xia et al., 2019).

Under this assumption, Patrini et al. observe that, for all $y, \widetilde{y} \in \mathcal{Y}$,

$$\widetilde{\eta}_{\widetilde{y}}(\bar{x}^y) = \left(\mathbf{C}^\top \boldsymbol{\eta}(\bar{x}^y)\right)_{\widetilde{y}} = \left(\mathbf{C}^\top \mathbf{e}_y\right)_{\widetilde{y}} = \gamma_{y,\widetilde{y}} \,.$$

Therefore, if one can identify such perfect examples/anchor points $\bar{x}^y$, and if the class probability estimates $\widehat{\widetilde{\boldsymbol{\eta}}}(x)$ are accurate, then one can estimate the noise rates via

$$\widehat{\gamma}_{y,\widetilde{y}} = \widehat{\widetilde{\eta}}_{\widetilde{y}}(\bar{x}^y) \quad \forall y, \widetilde{y} \in [n] \,.$$

As discussed previously, provided one has a sufficiently large training sample, accurate class probability estimates can be formed by minimizing a strongly proper composite surrogate over a suitably rich function class. The main step that is needed, therefore, is to identify the 'perfect' examples/anchor points $\bar{x}^y$ above.

Patrini et al. suggest identifying such anchor points by first estimating a CPE model $\widehat{\widetilde{\boldsymbol{\eta}}}$, and then taking a large collection of available instances $\mathcal{X}_{\text{train}} \subset \mathcal{X}$ drawn IID from the marginal μ (these could just be the training instances in $\widetilde{S}$ or could include other unlabeled instances as well), and estimating anchor points according to

$$\widehat{\bar{x}}^y \in \operatorname*{argmax}_{x \in \mathcal{X}_{\text{train}}} \widehat{\widetilde{\eta}}_y(x)$$

However, note that these anchor points should be chosen to maximize the *true* class probability $\eta_y(x)$, *not* the noisy class probability $\widetilde{\eta}_y(x)$! Therefore, the above method (also used by Yao et al., according to footnote 2 in their paper) is in general incorrect. Of course, we do have a relation between $\boldsymbol{\eta}$ and $\widetilde{\boldsymbol{\eta}}$ (Eq. (1)), but that relation involves $\mathbf{C}$; without knowledge of $\mathbf{C}$, we cannot in general use a noisy CPE model $\widehat{\widetilde{\boldsymbol{\eta}}}$ to find instances maximizing $\eta_y(x)$.

Nevertheless, surprisingly, Patrini et al. and Yao et al. did report some successful experiments with their methods. On investigating further, we identified a sufficient condition on the noise matrix $\mathbf{C}$ under which $\operatorname{argmax}_x \widetilde{\eta}_y(x) = \operatorname{argmax}_x \eta_y(x)$, and therefore, under which the above approach for estimating anchor points does work correctly, as well as a related necessary condition failing which the approach fails:

Theorem 5. *Suppose assumption (A) above holds.*

1. *If the noise matrix $\mathbf{C} = [\gamma_{y,\widetilde{y}}]$ satisfies the sufficient condition*
$$\gamma_{\widetilde{y},\widetilde{y}} > \gamma_{y,\widetilde{y}} \;\; \forall y \neq \widetilde{y} \,,$$
then provided that $\mathcal{X}_{train}$ is a large enough sample (drawn IID from μ) and the noisy class probabilities $\widetilde{\boldsymbol{\eta}}(x)$ are modeled accurately, the anchor point estimation method of Patrini et al. (2017) described above works correctly.
2. *If $\mathbf{C}$ fails to satisfy the necessary condition*
$$\gamma_{\widetilde{y},\widetilde{y}} \geq \gamma_{y,\widetilde{y}} \;\; \forall y \neq \widetilde{y} \,,$$
then the anchor point estimation method of Patrini et al. (2017) described above fails.

Algorithm 3 Iterative Noise Estimation Heuristic

Inputs:
(1) CPE model $\widehat{\widetilde{\boldsymbol{\eta}}} : \mathcal{X} \to \Delta_n$ (for noisy distribution)
(2) $\mathcal{X}_{\text{train}} \subset \mathcal{X}$
(3) Maximum number of iterations T
Initialize: $\widehat{\mathbf{C}}^{(1)} = \left[\widehat{\gamma}^{(1)}_{y,\widetilde{y}}\right] \leftarrow \mathbf{I}$
For $t = 1, \ldots, T$:
  $\forall y \in \mathcal{Y}: \ \widehat{\widetilde{x}}^y \leftarrow \operatorname{argmax}_{x \in \mathcal{X}_{\text{train}}} \left(\left((\widehat{\mathbf{C}}^{(t)})^\top \right)^{-1} \widehat{\widetilde{\boldsymbol{\eta}}}(x) \right)_y$
  $\forall y, \widetilde{y} \in \mathcal{Y}: \ \widehat{\gamma}^{(t+1)}_{y,\widetilde{y}} \leftarrow \widehat{\widetilde{\eta}}_{\widetilde{y}}(\widehat{\widetilde{x}}^y)$
  $\text{diff}^{(t)} \leftarrow \|\widehat{\mathbf{C}}^{(t+1)} - \widehat{\mathbf{C}}^{(t)}\|_F$
Output: $\widehat{\mathbf{C}}^{(t^*)}$, where $t^* = \operatorname{argmin}_{t \in [T]} \text{diff}^{(t)}$

It is worth noting that the noise matrices in Patrini et al.'s study that were estimated correctly by their method all satisfy the sufficient condition above; for the one noise matrix in their study which did not satisfy the necessary condition above, their estimation method failed (see Section 6.2 for details). Similarly, all the noise matrices considered in Yao et al.'s study satisfy the sufficient condition above; in our experiments, for noise matrices that fail to satisfy the necessary condition above, Yao et al.'s method also fails.

We also note that, in Patrini et al.'s study, after learning a noisy CPE model $\widehat{\widetilde{\boldsymbol{\eta}}}$ and estimating $\mathbf{C}$, a different learning algorithm that minimizes a noise-corrected loss was then used to learn a classifier $\widehat{h}$. In our case, after learning $\widehat{\widetilde{\boldsymbol{\eta}}}$ and estimating $\mathbf{C}$, we can simply output the plug-in classifier $\widehat{h} = \text{plug}^{\mathbf{L}}_{\widehat{\mathbf{C}}} \circ \widehat{\widetilde{\boldsymbol{\eta}}}$, with no additional training required.

5.2. An Iterative Noise Estimation Heuristic

Based on the discussion above, we propose an alternative, iterative noise estimation heuristic that aims to improve anchor point estimation, wherein we start with an estimate of $\widehat{\mathbf{C}} = \mathbf{I}$ (no noise), and iteratively feed in the current estimate into a corrected version of Patrini et al.'s method to obtain an updated estimate. The approach is shown in Algorithm 3. The first iteration simply corresponds to Patrini et al.'s original method; therefore, if $\mathbf{C}$ satisfies the condition of Theorem 5, then the first iteration produces an accurate estimate. Unfortunately, the method is not guaranteed to converge or to produce an accurate estimate in general; nevertheless, in our experiments, we find this method performs as well as, or better than, the methods of Patrini et al. and Yao et al. It remains an open question whether general noise matrices $\mathbf{C}$ can be estimated reliably using anchor points.

6. Experiments

We conducted two sets of experiments to evaluate our noise-corrected plug-in algorithm. In the first set of experiments, we generated synthetic data, and tested the sample complexity behavior of our algorithm, using linear models, for a variety of different noise matrices $\mathbf{C}$ with increasing values of $\|\mathbf{C}^{-1}\|_2$. In the second set of experiments, we compared the performance of our noise correction method with those of van Rooyen & Williamson (2017) and Patrini et al. (2017)), all using neural network models, on two real benchmark data sets; in this set of experiments, we used noise matrices $\mathbf{C}$ constructed for these data sets by Patrini et al. (2017), closely following their experimental settings. We also compared the performance of our noise estimation method with those of Patrini et al. (2017) and Yao et al. (2020). In all cases, we used the multiclass logistic loss (unmodified in our case, and modified as needed by each of the other algorithms). We summarize both sets of experiments below. In all cases, training labels were flipped randomly according to the prescribed (invertible) noise matrix $\mathbf{C}$; performance of the learned models was then measured on a clean test set.

6.1. Synthetic Data: Sample Complexity Behavior

In order to test the sample complexity behavior of our algorithm, we generated synthetic data from a known distribution (from which we could draw increasingly large training samples as needed). Specifically, we constructed a 5-class problem over a 10-dimensional instance space $\mathcal{X} = [-1, 1]^{10}$ as follows. Instances $\mathbf{x}$ were generated uniformly at random from $\mathcal{X}$. The class probability function $\boldsymbol{\eta} : \mathcal{X} \to \Delta_5$ was set to $\eta_y(\mathbf{x}) = \frac{\exp(\mathbf{w}_y^\top \mathbf{x})}{\sum_{y'=1}^{5} \exp(\mathbf{w}_{y'}^\top \mathbf{x})}$ for some fixed weight vectors $\mathbf{w}_1, \ldots, \mathbf{w}_5 \in \mathbb{R}^{10}$ (the entries of the weight vectors were drawn IID from $\mathcal{N}(0, 1)$ and then scaled so that $\|\mathbf{w}_y\|_2 = 1$). Given an instance $\mathbf{x}$, a clean label y was drawn randomly according to $\boldsymbol{\eta}(\mathbf{x})$. For any prescribed (row-stochastic) noise matrix $\mathbf{C}$, training labels y were then stochastically flipped to a noisy label $\widetilde{y}$ according to the probabilities in the y-th row of $\mathbf{C}$.

We tested the sample complexity behavior of our algorithm, implemented to minimize the multiclass logistic loss over linear models, for a variety of noise matrices $\mathbf{C}$ with increasing values of $\|\mathbf{C}^{-1}\|_2$.[7] We ran the algorithm on increasingly large (noisy) training samples (up to 40,000 examples) and measured the performance on a large test set of 10,000 (clean) examples. The results are shown in Figure 1: The left plot in the figure shows results for the 0-1 loss (shown as accuracy); the right plot shows results for a different target loss, specifically, the ordinal regression loss $\mathbf{L}^{\text{ord}}$ defined as $\ell^{\text{ord}}_{y,\widehat{y}} = |\widehat{y} - y|$.[8] We see that, as suggested by our regret transfer bound, as $\|\mathbf{C}^{-1}\|_2$ increases (i.e. as the matrix $\mathbf{C}$

[7] The implementation was in PyTorch (Paszke et al., 2019), and used the AdamW optimizer. The optimizer was run for 50 epochs over the training sample; the learning rate parameter was initially set to 0.01 and was halved at the end of every 5 epochs.

[8] Following Natarajan et al. (2013), for each noise matrix, we repeated each experiment 3 times with independent random corruptions of the training set using the same noise matrix; our results give the mean performance over the 3 runs.

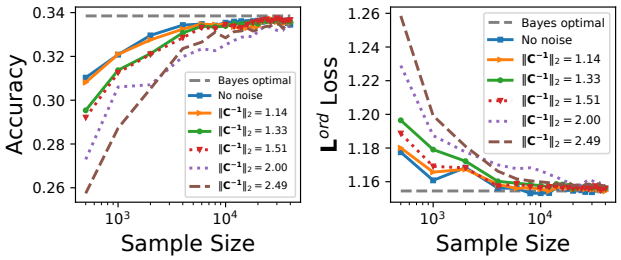

Figure 1. Sample complexity behavior of our algorithm on synthetic 5-class data for a variety of noise matrices $\mathbf{C}$ with increasing values of $\|\mathbf{C}^{-1}\|_2$. **Left:** 0-1 loss (shown as accuracy). **Right:** Ordinal regression loss $\mathbf{L}^{\text{ord}}$. As suggested by our regret bounds, as $\|\mathbf{C}^{-1}\|_2$ increases, the sample size needed to reach a given level of performance generally increases. See Section 6.1 for details.

becomes closer to being singular), the sample size required to achieve a given level of performance generally increases.[9]

6.2. Real Data: Comparison with Other Algorithms

We conducted experiments on several real data sets. Here we describe experiments on two benchmark data sets, MNIST (Lecun et al., 1998) and CIFAR10 (Krizhevsky & Hinton, 2009), where we compared our algorithm with the *unbiased estimator* method of van Rooyen & Williamson (2017) and the *forward* method of Patrini et al. (2017), all using neural network models, and also tested the incorporation of noise estimation methods. These experiments were designed to closely mimic experiments of Patrini et al. (2017); we used code provided by the authors[10] and kept the neural network architectures and all parameters as given.[11]

Both MNIST and CIFAR10 are 10-class data sets (see the supplementary material for details of the data sets). The experiments used 0-1 loss (measured as accuracy). In both cases, experiments were conducted with clean data (no noise) and with 6 noise matrices $\mathbf{C}$. One of these, $\mathbf{C}^{\text{sym}(0.2)}$, was a symmetric noise matrix with the following structure: all diagonal entries γ_{yy} were set to $1-\gamma$, where $\gamma = 0.2$; all off-diagonal entries were set to $\frac{\gamma}{n-1}$ (here $n = 10$). For such symmetric noise matrices (with $\gamma < \frac{n-1}{n}$) and 0-1 loss, it is known that no noise correction is needed, and that standard algorithms designed to learn a good classifier for 0-1 loss (on the noisy data) work correctly (van Rooyen & Williamson, 2017; Ghosh et al., 2017). The other 5 noise matrices were asymmetric, and were artificially designed by Patrini et al. (2017) to simulate some of the possible structures of real label noise, where a label might be replaced with some probability γ by some other similar label, for example, `Cat` $\rightarrow$ `Dog`. For each of MNIST and CIFAR10, Patrini et al. specified a set of such 'label noise' transitions to create specific parametric noise matrices $\mathbf{C}^{\text{MNIST}(\gamma)}$ and $\mathbf{C}^{\text{CIFAR10}(\gamma)}$, and instantiated these with $\gamma = 0.2, 0.6$; we additionally included $\gamma = 0.45, 0.55, 0.65$. The noise matrices $\mathbf{C}^{\text{sym}(0.2)}$ and $\mathbf{C}^{\text{MNIST}(\gamma)}$, $\mathbf{C}^{\text{CIFAR10}(\gamma)}$ for $\gamma < 0.5$ all satisfy the sufficient condition of Theorem 5; the matrices $\mathbf{C}^{\text{MNIST}(\gamma)}$, $\mathbf{C}^{\text{CIFAR10}(\gamma)}$ for $\gamma > 0.5$ fail to satisfy the necessary condition. Details of these noise matrices, as well as the neural network models used and associated parameter settings, can be found in the supplementary material.

The results are summarized in Tables 1 and 2, respectively. For each algorithm, we implemented four versions: one with the noise matrix $\mathbf{C}$ known, and the other three with the noise matrix estimated using either the method of Patrini et al. (2017) (denoted $\widehat{\mathbf{C}}^{\text{Patrini}}$), the Dual T method of Yao et al. (2020) ($\widehat{\mathbf{C}}^{\text{DT}}$), or our iterative noise estimation heuristic ($\widehat{\mathbf{C}}^{\text{iter}}$).[12] Several observations are in order. First, for the symmetric noise matrix $\mathbf{C}^{\text{sym}(0.2)}$, standard logistic regression with no noise correction does well as expected; for heavy asymmetric noise ($\mathbf{C}^{\text{MNIST}(\gamma)}$ and $\mathbf{C}^{\text{CIFAR10}(\gamma)}$ for $\gamma > 0.5$), standard logistic regression without noise correction does not do well. Second, our noise-corrected plug-in method is comparable to the other noise-corrected methods, even though it requires no change to the training process. Third, our iterative noise estimation heuristic either performs similarly to the noise estimation methods of Patrini et al. and Yao et al., or in some cases (particularly $\mathbf{C}^{\text{MNIST}(\gamma)}$ for $\gamma > 0.5$) significantly outperforms their methods. Finally, for the noise matrices that satisfy the sufficient condition of Theorem 5, all three noise estimation methods perform well; for the noise matrices that fail to satisfy the necessary condition, no method achieves perfect estimation.

It is worth pointing out again that all three noise estimation methods (the methods of Patrini et al. (2017) and Yao et al. (2020), and our iterative method) make use of a noisy CPE model $\widehat{\widetilde{\boldsymbol{\eta}}}(x)$ learned from the noisy training data. Our noise-corrected plug-in algorithm makes use of this noisy CPE model directly, simply applying a noise-corrected plug-in step at prediction time, and does not need any further re-training; on the other hand, the other two noise correc-

[9] We note that for the synthetic data distribution described above, although the clean class probabilities $\boldsymbol{\eta}(\mathbf{x})$ take the form of a softmax-of-linear model, the noisy class probabilities $\widetilde{\boldsymbol{\eta}}(\mathbf{x})$ are not of this form. Therefore, even though the plots in Figure 1 seem to suggest our algorithm converges to the Bayes optimal performance, strictly speaking, this is not the case: The algorithm does appear to have learned a fairly accurate model for the noisy class probabilities $\widetilde{\boldsymbol{\eta}}(\mathbf{x})$, but it cannot express them exactly; in order to truly model them exactly, we would need to implement the algorithm using a richer function class. (We do not do this here since the difference in performance would be unnoticeable. We use richer function classes in the experiments with real data, where we employ neural network models.)

[10] https://github.com/giorgiop/loss-correction

[11] We note that for some parameters (e.g. batch size), there is a discrepancy between the settings used in the code and those mentioned in the paper; we used the settings in the code.

[12] Our iterative noise estimation heuristic was implemented with maximum number of iterations T set to 1000.

Table 1. Test accuracy (percentage) on MNIST data, shown as the mean (with standard error of the mean in parentheses) over 5 random trials. In each column, the best algorithm(s) using the known noise matrix $\mathbf{C}$ and the best algorithm(s) using each of the 3 noise estimation methods (Patrini et al., Dual T, and our iterative heuristic) are shown in bold font; among the latter, the best algorithm + noise estimation combination overall is further enclosed in asterisks. See Section 6.2 for details.

Algorithm	**No noise**	$\mathbf{C}^{\text{sym}(0.2)}$	$\mathbf{C}^{\text{MNIST}(0.2)}$	$\mathbf{C}^{\text{MNIST}(0.45)}$	$\mathbf{C}^{\text{MNIST}(0.55)}$	$\mathbf{C}^{\text{MNIST}(0.6)}$	$\mathbf{C}^{\text{MNIST}(0.65)}$
Logistic	**92.84** (0.14)	**92.00** (0.07)	91.58 (0.08)	80.80 (0.30)	58.27 (0.27)	52.08 (0.23)	49.86 (0.05)
Unbiased, $\mathbf{C}$	92.76 (0.02)	91.98 (0.11)	**92.24** (0.08)	**89.75** (0.30)	**89.54** (0.11)	**90.72** (0.06)	**90.68** (0.08)
Forward, $\mathbf{C}$	**92.84** (0.06)	91.69 (0.08)	92.00 (0.09)	84.52 (1.48)	82.14 (2.48)	87.47 (1.43)	89.57 (0.92)
Plug-in, $\mathbf{C}$	**92.84** (0.14)	**92.00** (0.08)	92.05 (0.10)	87.57 (0.46)	87.70 (0.18)	89.31 (0.19)	89.23 (0.06)
Unbiased, $\widehat{\mathbf{C}}^{\text{Patrini}}$	92.63 (0.05)	91.45 (0.05)	91.94 (0.02)	**88.50** (0.28)	68.50 (3.41)	66.91 (2.11)	69.45 (0.94)
Forward, $\widehat{\mathbf{C}}^{\text{Patrini}}$	92.68 (0.10)	91.75 (0.05)	***92.20*** (0.09)	83.44 (1.90)	71.57 (2.18)	68.14 (3.01)	61.58 (2.22)
Plug-in, $\widehat{\mathbf{C}}^{\text{Patrini}}$	**92.84** (0.13)	**92.01** (0.01)	91.97 (0.09)	87.01 (0.54)	**73.17** (1.37)	**70.37** (1.09)	**69.64** (0.97)
Unbiased, $\widehat{\mathbf{C}}^{\text{DT}}$	92.16 (0.04)	91.29 (0.07)	91.50 (0.09)	85.79 (0.59)	60.30 (1.70)	53.72 (1.10)	52.21 (4.55)
Forward, $\widehat{\mathbf{C}}^{\text{DT}}$	92.37 (0.07)	91.46 (0.06)	91.80 (0.07)	80.77 (2.37)	62.84 (3.77)	**60.79** (2.94)	**58.15** (2.28)
Plug-in, $\widehat{\mathbf{C}}^{\text{DT}}$	**92.84** (0.13)	**91.92** (0.04)	**92.04** (0.08)	**86.51** (0.60)	**66.07** (2.62)	58.19 (0.43)	57.99 (3.13)
Unbiased, $\widehat{\mathbf{C}}^{\text{iter}}$	92.73 (0.07)	91.76 (0.08)	92.00 (0.07)	***89.51*** (0.19)	74.95 (0.22)	71.24 (3.52)	70.98 (3.44)
Forward, $\widehat{\mathbf{C}}^{\text{iter}}$	92.70 (0.07)	91.60 (0.17)	**92.16** (0.04)	85.22 (2.64)	***81.36*** (1.26)	***73.52*** (4.57)	***74.42*** (4.35)
Plug-in, $\widehat{\mathbf{C}}^{\text{iter}}$	**92.85** (0.13)	***92.03*** (0.04)	91.96 (0.09)	87.55 (0.46)	76.96 (0.13)	71.65 (3.26)	70.68 (3.19)

Table 2. Test accuracy (percentage) on CIFAR10 data, shown as the mean (with standard error of the mean in parentheses) over 5 random trials. In each column, the best algorithm(s) using the known noise matrix $\mathbf{C}$ and the best algorithm(s) using each of the 3 noise estimation methods (Patrini et al., Dual T, and our iterative heuristic) are shown in bold font; among the latter, the best algorithm + noise estimation combination overall is further enclosed in asterisks. See Section 6.2 for details.

Algorithm	**No noise**	$\mathbf{C}^{\text{sym}(0.2)}$	$\mathbf{C}^{\text{CIFAR10}(0.2)}$	$\mathbf{C}^{\text{CIFAR10}(0.45)}$	$\mathbf{C}^{\text{CIFAR10}(0.55)}$	$\mathbf{C}^{\text{CIFAR10}(0.6)}$	$\mathbf{C}^{\text{CIFAR10}(0.65)}$
Logistic	**89.76** (0.07)	85.26 (0.16)	86.95 (0.12)	76.56 (0.31)	63.78 (0.58)	58.1 (0.39)	54.79 (0.31)
Unbiased, $\mathbf{C}$	89.6 (0.08)	81.82 (0.27)	84.1 (0.14)	61.91 (2.32)	57.43 (3.81)	70.12 (2.43)	74.91 (1.99)
Forward, $\mathbf{C}$	89.6 (0.14)	**86.62** (0.13)	**88.7** (0.16)	**85.71** (0.2)	**85.12** (0.11)	**86.99** (0.03)	**87.12** (0.14)
Plug-in, $\mathbf{C}$	**89.76** (0.07)	85.26 (0.16)	87.46 (0.09)	82.71 (0.26)	81.8 (0.24)	83.5 (0.12)	84.01 (0.16)
Unbiased, $\widehat{\mathbf{C}}^{\text{Patrini}}$	89.54 (0.16)	82.27 (0.14)	86.08 (0.42)	69.45 (1.75)	67.15 (1.97)	69.77 (0.58)	69.94 (0.45)
Forward, $\widehat{\mathbf{C}}^{\text{Patrini}}$	89.66 (0.05)	**86.05** (0.23)	**88.11** (0.06)	**84.67** (0.48)	***78.78*** (1.14)	75.47 (0.36)	***74.48*** (0.48)
Plug-in, $\widehat{\mathbf{C}}^{\text{Patrini}}$	**89.76** (0.07)	85.29 (0.19)	87.39 (0.08)	82.46 (0.24)	78.02 (0.24)	**75.48** (0.19)	74.26 (0.3)
Unbiased, $\widehat{\mathbf{C}}^{\text{DT}}$	89.3 (0.16)	82.7 (0.12)	83.43 (0.23)	67.67 (2.05)	63.38 (0.54)	63.4 (1.05)	62.64 (1.16)
Forward, $\widehat{\mathbf{C}}^{\text{DT}}$	89.75 (0.21)	***86.93*** (0.08)	***88.32*** (0.1)	***84.72*** (0.42)	**75.98** (0.48)	69.85 (1.68)	61.89 (0.6)
Plug-in, $\widehat{\mathbf{C}}^{\text{DT}}$	**89.77** (0.07)	85.14 (0.19)	87.39 (0.1)	81.91 (0.32)	75.6 (0.29)	**71.21** (0.34)	**68.04** (0.7)
Unbiased, $\widehat{\mathbf{C}}^{\text{iter}}$	89.67 (0.06)	82.09 (0.14)	86.0 (0.19)	68.75 (2.68)	65.85 (1.23)	68.77 (1.1)	67.83 (1.65)
Forward, $\widehat{\mathbf{C}}^{\text{iter}}$	89.5 (0.07)	**85.92** (0.2)	**88.17** (0.03)	**84.09** (0.71)	77.64 (0.73)	***75.71*** (0.32)	**74.46** (0.59)
Plug-in, $\widehat{\mathbf{C}}^{\text{iter}}$	**89.76** (0.07)	85.27 (0.2)	87.4 (0.08)	82.41 (0.22)	**77.94** (0.19)	75.49 (0.17)	74.27 (0.29)

tion methods above both need to further minimize a noise-corrected loss on the noisy data in order to learn a classifier.

7. Conclusion

We have provided a simple noise-corrected plug-in method for general multiclass class-conditional label noise (CCN) that requires no change to the training process. Noise correction takes place at prediction time, and after a one-time matrix inversion and multiplication step, requires $O(n^2)$ time per prediction, where n is the number of classes. For general loss matrices $\mathbf{L}$, this is the same computational cost that is needed for standard plug-in methods; for the 0-1 loss, it is a factor of n larger than the standard cost (for small to moderate n, this may still be a smaller cost overall as compared to the cost of modifying the training process). We have also provided quantitative regret transfer bounds for our method that quantify the effect of learning from noisy labels, as well as an iterative noise estimation heuristic.

One possible issue to be careful about is that accurate estimation of noisy class probabilities can potentially be challenging due to their typically higher variance (particularly with neural networks, which often exhibit high calibration errors (Guo et al., 2017; Rahimi et al., 2020)) – while we did not find this to be a significant concern in our experiments, it could possibly be an issue for certain types of data sets or noise. It remains an open question whether general noise matrices $\mathbf{C}$ can be estimated reliably using anchor points.

Acknowledgments

Thanks to the anonymous reviewers for several helpful comments and pointers. This material is based upon work supported in part by the US National Science Foundation (NSF) under Grant Nos. 1934876 and 1717290. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Agarwal, S. Surrogate regret bounds for bipartite ranking via strongly proper losses. *Journal of Machine Learning Research*, 15:1653–1674, 2014.

Angluin, D. and Laird, P. D. Learning from noisy examples. *Mach. Learn.*, 2(4):343–370, 1987.

Aslam, J. A. and Decatur, S. E. On the sample complexity of noise-tolerant learning. *Inf. Process. Lett.*, 57(4):189–195, 1996.

Blum, A. and Mitchell, T. M. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT 1998*, pp. 92–100. ACM, 1998.

Bylander, T. Learning linear threshold functions in the presence of classification noise. In *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory, COLT 1994*, pp. 340–347. ACM, 1994.

Cesa-Bianchi, N., Dichterman, E., Fischer, P., Shamir, E., and Simon, H. U. Sample-efficient strategies for learning in the presence of noise. *J. ACM*, 46(5):684–719, 1999.

Cheng, J., Liu, T., Ramamohanarao, K., and Tao, D. Learning with bounded instance and label-dependent label noise. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1789–1799. PMLR, 2020.

Duchi, J., Mackey, L., and Jordan, M. On the consistency of ranking algorithms. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

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

Ghosh, A., Kumar, H., and Sastry, P. S. Robust loss functions under label noise for deep neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 2017*, pp. 1919–1925. AAAI Press, 2017.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778. IEEE Computer Society, 2016.

Kearns, M. J. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, 1998.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

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. doi: 10.1109/5.726791.

Menon, A. K., van Rooyen, B., Ong, C. S., and Williamson, B. Learning from corrupted binary labels via class-probability estimation. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 125–134. JMLR.org, 2015.

Menon, A. K., van Rooyen, B., and Natarajan, N. Learning from binary labels with instance-dependent noise. *Mach. Learn.*, 107(8-10):1561–1595, 2018.

Natarajan, N., Dhillon, I. S., Ravikumar, P., and Tewari, A. Learning with noisy labels. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013*, pp. 1196–1204, 2013.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Patrini, G., Rozza, A., Menon, A. K., Nock, R., and Qu, L. Making deep neural networks robust to label noise: A loss correction approach. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 2233–2241. IEEE Computer Society, 2017.

Rahimi, A., Shaban, A., Cheng, C., Hartley, R., and Boots, B. Intra order-preserving functions for calibration of multi-class neural networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.

Song, H., Kim, M., Park, D., and Lee, J. Learning from noisy labels with deep neural networks: A survey. *CoRR*, abs/2007.08199, 2020.

van Rooyen, B. and Williamson, R. C. A theory of learning with corrupted labels. *J. Mach. Learn. Res.*, 18:228:1–228:50, 2017.

Wang, R., Liu, T., and Tao, D. Multiclass learning with partially corrupted labels. *IEEE Trans. Neural Networks Learn. Syst.*, 29(6):2568–2580, 2018.

Williamson, R. C., Vernet, E., and Reid, M. D. Composite multiclass losses. *J. Mach. Learn. Res.*, 17:223:1–223:52, 2016.

Xia, X., Liu, T., Wang, N., Han, B., Gong, C., Niu, G., and Sugiyama, M. Are anchor points really indispensable in label-noise learning? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pp. 6835–6846, 2019.

Yao, Y., Liu, T., Han, B., Gong, M., Deng, J., Niu, G., and Sugiyama, M. Dual T: reducing estimation error for transition matrix in label-noise learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.

Learning from Noisy Labels with No Change to the Training Process

Supplementary Material

A. Proof of Theorem 1

Proof. We use $\langle \cdot, \cdot \rangle$ to denote the standard inner product.

$$
\begin{aligned}
&\operatorname{regret}_D^{\mathbf{L}}[\,\widehat{h}\,] \\
&= \mathbf{E}_X\left[\langle \boldsymbol{\eta}(X), \boldsymbol{\ell}_{\widehat{h}(X)}\rangle - \min_{y\in[n]} \langle \boldsymbol{\eta}(X), \boldsymbol{\ell}_y\rangle\right] \\
&= \mathbf{E}_X\left[\max_{y\in[n]} \langle \boldsymbol{\eta}(X), \boldsymbol{\ell}_{\widehat{h}(X)} - \boldsymbol{\ell}_y\rangle\right] \\
&= \mathbf{E}_X\left[\max_{y\in[n]} \langle (\mathbf{C}^\top)^{-1}\widetilde{\boldsymbol{\eta}}(X), \boldsymbol{\ell}_{\widehat{h}(X)} - \boldsymbol{\ell}_y\rangle\right] \\
&= \mathbf{E}_X\left[\max_{y\in[n]} \langle \widetilde{\boldsymbol{\eta}}(X), \mathbf{C}^{-1}(\boldsymbol{\ell}_{\widehat{h}(X)} - \boldsymbol{\ell}_y)\rangle\right] \quad \text{(by property of adjoint)} \\
&\leq \mathbf{E}_X\left[\max_{y\in[n]} \langle \widetilde{\boldsymbol{\eta}}(X) - \widehat{\widetilde{\boldsymbol{\eta}}}(X), \mathbf{C}^{-1}(\boldsymbol{\ell}_{\widehat{h}(X)} - \boldsymbol{\ell}_y)\rangle\right] \\
&\qquad \text{(since by the definition of } \widehat{h}(X),\ \langle \widehat{\widetilde{\boldsymbol{\eta}}}(X), \mathbf{C}^{-1}(\boldsymbol{\ell}_{\widehat{h}(X)} - \boldsymbol{\ell}_y)\rangle \leq 0\ \forall y \in [n]) \\
&\leq \mathbf{E}_X\left[\left\|\widehat{\widetilde{\boldsymbol{\eta}}}(X) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2 \cdot \left\|\mathbf{C}^{-1}\right\|_2 \cdot \max_{y\in[n]} \left\|\boldsymbol{\ell}_{\widehat{h}(X)} - \boldsymbol{\ell}_y\right\|_2\right] \\
&\qquad \text{(by Cauchy-Schwarz inequality)} \\
&\leq 2\max_{y\in[n]} \left\|\boldsymbol{\ell}_y\right\|_2 \cdot \left\|\mathbf{C}^{-1}\right\|_2 \cdot \mathbf{E}_X\left[\left\|\widehat{\widetilde{\boldsymbol{\eta}}}(X) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2\right]
\end{aligned}
$$

□

B. Proof of Theorem 4

Proof. By Theorem 1, we have

$$
\operatorname{regret}_D^{\mathbf{L}}[\,\widehat{h}\,] \leq 2\max_y \left\|\boldsymbol{\ell}_y\right\|_2 \cdot \left\|\mathbf{C}^{-1}\right\|_2 \cdot \mathbf{E}_X\left[\left\|\widehat{\widetilde{\boldsymbol{\eta}}}(X) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2\right]. \tag{3}
$$

Then, since ψ is s-strongly proper composite with link function $\boldsymbol{\lambda}$, we have

$$
\begin{aligned}
&\operatorname{regret}_D^{\psi}[\,\widehat{\widetilde{\mathbf{f}}}\,] \\
&= \mathbf{E}_X\left[\mathbf{E}_{Y|X\sim\widetilde{\boldsymbol{\eta}}(X)}\left[\psi(Y, \widehat{\widetilde{\mathbf{f}}}(X))\right] - \inf_{\mathbf{u}\in\mathbb{R}^{n-1}} \mathbf{E}_{Y|X\sim\widetilde{\boldsymbol{\eta}}(X)}\left[\psi(Y, \mathbf{u})\right]\right] \\
&= \mathbf{E}_X\left[\mathbf{E}_{Y|X\sim\widetilde{\boldsymbol{\eta}}(X)}\left[\psi(Y, \widehat{\widetilde{\mathbf{f}}}(X)) - \psi(Y, \boldsymbol{\lambda}(\widetilde{\boldsymbol{\eta}}(X)))\right]\right] \\
&\qquad \text{(by definition of strongly proper composite multiclass loss)} \\
&\geq \mathbf{E}_X\left[\frac{s}{2}\left\|\boldsymbol{\lambda}^{-1}(\widehat{\widetilde{\mathbf{f}}}(X)) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2^2\right] \\
&= \frac{s}{2}\mathbf{E}_X\left[\left\|\widehat{\widetilde{\boldsymbol{\eta}}}(X) - \widetilde{\boldsymbol{\eta}}(X)\right\|_2^2\right]
\end{aligned} \tag{4}
$$

Combining Eqs. (3, 4), and applying Jensen's inequality (to the convex function $x \mapsto x^2$) establishes the result.

□

C. Proof of Lemma 3

Proof. We will show for all $\mathbf{p} \in \Delta_n$ and $\mathbf{u} \in \mathbb{R}^{n-1}$,

$$\mathbf{E}_{Y\sim\mathbf{p}}\Big[\psi_{\text{mlog}}(Y,\mathbf{u}) - \psi_{\text{mlog}}(Y,\boldsymbol{\lambda}_{\text{mlog}}(\mathbf{p}))\Big] \geq \frac{1}{2}\big\|\boldsymbol{\lambda}_{\text{mlog}}^{-1}(\mathbf{u}) - \mathbf{p}\big\|_2^2 .$$

Fix $\mathbf{p} \in \Delta_n$ and $\mathbf{u} \in \mathbb{R}^{n-1}$. Then

$$\begin{aligned}
&\mathbf{E}_{Y\sim\mathbf{p}}\Big[\psi_{\text{mlog}}(Y,\mathbf{u}) - \psi_{\text{mlog}}(Y,\boldsymbol{\lambda}_{\text{mlog}}(\mathbf{p}))\Big] \\
&= -\sum_{i\in[n]} p_i \ln\left((\boldsymbol{\lambda}_{\text{mlog}}^{-1}(\mathbf{u}))_i\right) + \sum_{i\in[n]} p_i \ln(p_i) \\
&= \sum_{i\in[n]} p_i \ln\Big(\frac{p_i}{(\boldsymbol{\lambda}_{\text{mlog}}^{-1}(\mathbf{u}))_i}\Big) \\
&= D_{KL}(\mathbf{p}||\boldsymbol{\lambda}_{\text{mlog}}^{-1}(\mathbf{u})) \quad \text{by the definition of Kullback-Leibler divergence} \\
&\geq \frac{1}{2}\big\|\mathbf{p} - \boldsymbol{\lambda}_{\text{mlog}}^{-1}(\mathbf{u})\big\|_1^2 \quad \text{using Pinsker's inequality and properties of total variation distance} \\
&\geq \frac{1}{2}\big\|\mathbf{p} - \boldsymbol{\lambda}_{\text{mlog}}^{-1}(\mathbf{u})\big\|_2^2 .
\end{aligned}$$

□

D. Proof of Theorem 5

Proof. **Part 1 (Sufficiency).**

Suppose $\mathbf{C}$ satisfies the given sufficient condition, i.e. that

$$\gamma_{\widetilde{y},\widetilde{y}} > \gamma_{y,\widetilde{y}} \;\; \forall y \neq \widetilde{y} .$$

We will show that

$$\operatorname*{argmax}_x \eta_y(x) = \operatorname*{argmax}_x \widetilde{\eta}_y(x) \;\; \forall y \in [n] ;$$

the claim will then follow.

Fix any class $y \in [n]$.

First, suppose $x' \in \operatorname{argmax}_x \eta_y(x)$. Then by assumption (A), it must be the case that $\eta_y(x') = 1$, i.e. that $\boldsymbol{\eta}(x') = \mathbf{e}_y$. This gives

$$\widetilde{\eta}_y(x') = (\mathbf{C}^\top \boldsymbol{\eta}(x'))_y = (\mathbf{C}^\top \mathbf{e}_y)_y = \gamma_{y,y} .$$

Now for any $x \in \mathcal{X}$, we have

$$\widetilde{\eta}_y(x) = (\mathbf{C}^\top \boldsymbol{\eta}(x))_y = \sum_{y'=1}^n \gamma_{y',y}\eta_{y'}(x) \leq \sum_{y'=1}^n \gamma_{y,y}\eta_{y'}(x) = \gamma_{y,y} = \widetilde{\eta}_y(x') .$$

Thus $x' \in \operatorname{argmax}_x \widetilde{\eta}_y(x)$. This establishes $\operatorname{argmax}_x \eta_y(x) \subseteq \operatorname{argmax}_x \widetilde{\eta}_y(x)$.

Conversely, suppose $x' \in \operatorname{argmax}_x \widetilde{\eta}_y(x) = \operatorname{argmax}_x (\mathbf{C}^\top \boldsymbol{\eta}(x))_y$. This means

$$\sum_{y'=1}^n \gamma_{y',y}\eta_{y'}(x') \geq \sum_{y'=1}^n \gamma_{y',y}\eta_{y'}(x) \;\; \forall x \in \mathcal{X} .$$

By assumption (A), there exists $\bar{x}^y \in \mathcal{X}$ such that $\boldsymbol{\eta}(\bar{x}^y) = \mathbf{e}_y$. Applying the above inequality to $x = \bar{x}^y$, we have

$$\sum_{y'=1}^n \gamma_{y',y}\eta_{y'}(x') \geq \sum_{y'=1}^n \gamma_{y',y}\eta_{y'}(\bar{x}^y) = \gamma_{y,y} .$$

Moreover, we have

$$\sum_{y'=1}^{n} \gamma_{y',y} \eta_{y'}(x') \leq \gamma_{y,y} \,.$$

Combining the above two inequalities, we get

$$\sum_{y'=1}^{n} \gamma_{y',y} \eta_{y'}(x') = \gamma_{y,y} \,.$$

Since $\gamma_{y',y} < \gamma_{y,y}$ for all $y' \neq y$, this means we must have $\boldsymbol{\eta}(x') = \mathbf{e}_y$. Thus, $x' \in \operatorname{argmax}_x \eta_y(x)$. This establishes $\operatorname{argmax}_x \widetilde{\eta}_y(x) \subseteq \operatorname{argmax}_x \eta_y(x)$.

Part 2 (Necessity).

Suppose that $\mathbf{C}$ fails to satisfy the given necessary condition, i.e. that there exist $y \neq \widetilde{y}$ such that

$$\gamma_{\widetilde{y},\widetilde{y}} < \gamma_{y,\widetilde{y}} \,.$$

We will show that $\operatorname{argmax}_x \eta_{\widetilde{y}}(x) \neq \operatorname{argmax}_x \widetilde{\eta}_{\widetilde{y}}(x)$.

We give a proof by contradiction. In particular, let if possible $\operatorname{argmax}_x \eta_{\widetilde{y}}(x) = \operatorname{argmax}_x \widetilde{\eta}_{\widetilde{y}}(x) = \operatorname{argmax}_x (\mathbf{C}^\top \boldsymbol{\eta}(x))_{\widetilde{y}}$.

By assumption (A), there exists $\bar{x}^{\widetilde{y}} \in \mathcal{X}$ such that $\boldsymbol{\eta}(\bar{x}^{\widetilde{y}}) = \mathbf{e}_{\widetilde{y}}$, so this means $\bar{x}^{\widetilde{y}} \in \operatorname{argmax}_x \eta_{\widetilde{y}}(x) = \operatorname{argmax}_x \widetilde{\eta}_{\widetilde{y}}(x) = \operatorname{argmax}_x (\mathbf{C}^\top \boldsymbol{\eta}(x))_{\widetilde{y}}$. This means

$$\gamma_{\widetilde{y},\widetilde{y}} = \sum_{y'=1}^{n} \gamma_{y',\widetilde{y}} \eta_{y'}(\bar{x}^{\widetilde{y}}) \geq \sum_{y'=1}^{n} \gamma_{y',\widetilde{y}} \eta_{y'}(x) \quad \forall x \in \mathcal{X} \,.$$

But by assumption (A), we can also find $\bar{x}^{y} \in \mathcal{X}$ such that $\boldsymbol{\eta}(\bar{x}^{y}) = \mathbf{e}_y$. Applying the above inequality to $x = \bar{x}^{y}$ then gives

$$\gamma_{\widetilde{y},\widetilde{y}} \geq \sum_{y'=1}^{n} \gamma_{y',\widetilde{y}} \eta_{y'}(\bar{x}^{y}) = \gamma_{y,\widetilde{y}} \,,$$

contradicting our assumption. Therefore, we must have $\operatorname{argmax}_x \eta_{\widetilde{y}}(x) \neq \operatorname{argmax}_x \widetilde{\eta}_{\widetilde{y}}(x)$. □

E. Additional Experimental Details

Table 3. Details of MNIST and CIFAR10 data sets.

Data set	**# train**	**# test**	**# classes** (n)	**# features** (d)
MNIST	60,000	10,000	10	784
CIFAR10	50,000	10,000	10	3072

For MNIST, the asymmetric noise matrix $\mathbf{C}^{\text{MNIST}(\gamma)}$ includes the following label noise transitions: $2 \to 7$, $3 \to 8$, $5 \leftrightarrow 6$, $7 \to 1$. Following Patrini et al. (2017), features were normalized to $[0, 1]$, and two fully connected hidden layers of size 128 were trained, with ReLU activation and dropout rate 0.2.[13]

For CIFAR10, the asymmetric noise matrix $\mathbf{C}^{\text{CIFAR10}(\gamma)}$ includes the following label noise transitions: `Truck` $\to$ `Automobile`, `Bird` $\to$ `Airplane`, `Deer` $\to$ `Horse`, `Cat` $\leftrightarrow$ `Dog`. Again following Patrini et al. (2017), per-pixel mean subtraction and data augmentation were performed, and a 14-layer residual network (ResNet) (He et al., 2016) was trained.[14]

[13] Batch size was 32. AdaGrad (Duchi et al., 2010) was run for 40 epochs with default parameters.

[14] Batch size was 32. SGD was run for 120 epochs with momentum 0.9 and learning rate set to 0.1 initially and divided by 10 after 40 and 80 epochs; weight decay was 10^{-4}.