Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels

Yikai Wang Xinwei Sun Yanwei Fu*
School of Data Science, Fudan University
{yikaiwang19, sunxinwei, yanweifu}@fudan.edu.cn

Abstract

Noisy training set usually leads to the degradation of generalization and robustness of neural networks. In this paper, we propose using a theoretically guaranteed noisy label detection framework to detect and remove noisy data for Learning with Noisy Labels (LNL). Specifically, we design a penalized regression to model the linear relation between network features and one-hot labels, where the noisy data are identified by the non-zero mean shift parameters solved in the regression model. To make the framework scalable to datasets that contain a large number of categories and training data, we propose a split algorithm to divide the whole training set into small pieces that can be solved by the penalized regression in parallel, leading to the Scalable Penalized Regression (SPR) framework. We provide the non-asymptotic probabilistic condition for SPR to correctly identify the noisy data. While SPR can be regarded as a sample selection module for standard supervised training pipeline, we further combine it with semi-supervised algorithm to further exploit the support of noisy data as unlabeled data. Experimental results on several benchmark datasets and real-world noisy datasets show the effectiveness of our framework. Our code and pretrained models are released at https://github.com/Yikai-Wang/SPR-LNL.

1. Introduction

Deep learning has achieved remarkable success on many topics of supervised learning with millions of labeled training data. The performance heavily relies on the quality of label annotation since neural networks are susceptible to noisy labels and even can easily memorize randomly labeled annotations [63], leading to the degradation of generalization and robustness. In many real-world scenarios, it is expensive and difficult to obtain precise labels, exposing a realistic challenge for supervised deep models to learn with noisy data.

There is a large literature for this challenge from various perspectives, including modifying the network architectures [6, 12, 13, 59] or loss functions [11, 27, 53, 65], or dynamically selecting clean data during training [5, 14, 17, 27, 34, 40, 44, 61]. Particularly, the dynamic sample selection methods adopt the spirit of providing only clean data for the training. Such a spirit can form a 'virtuous' cycle between the noisy data elimination and network training: the elimination of noisy data can help the network training; and on the other hand, the improved network is empowered with a better ability in picking up clean data. As this virtuous cycle evolves, the performance can be improved.

Typical principles to identify outliers include large loss [14], inconsistent prediction [67], and irregular feature representation [57]. The former two principles focus on the label space, while the last one focuses on the feature space of the same class. In this paper, we unify the label and feature space and assume linear relationship between the feature-label pair (denoted as $(\boldsymbol{x}_i, \boldsymbol{y}_i)$) of data i by

$$\boldsymbol{y}_i = \boldsymbol{x}_i^\top \boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{1}$$

where $\boldsymbol{x}_i \in \mathbb{R}^p$ is the feature vector, and $\boldsymbol{y}_i \in \mathbb{R}^c$ is the one-hot label vector; $\boldsymbol{\beta} \in \mathbb{R}^{p \times c}$ is the fixed (unknown) coefficient matrix and $\boldsymbol{\varepsilon} \in \mathbb{R}^c$ is random noise. This linear relation is approximately established as the networks are trained to minimize the divergence between a (soft-max) linear projection of the feature and one-hot label vector. For a well-trained network, the output prediction of clean data is expected to be as similar to a one-hot vector as possible, while for noisy data the output is dense. Intuitively, when the linear relation is well-approximated without soft-max operation, the corresponding data is likely to be clean data.

The simplest way to identify the suspected outliers in the linear model is checking the predict error, or residual, $\boldsymbol{r}_i = \boldsymbol{y}_i - \boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}$ is the estimate of $\boldsymbol{\beta}$. The larger $\|\boldsymbol{r}\|$ indicates more possibility for the instance i to be outlier/noisy data. The classical statistical method to test whether the instance $\boldsymbol{r}_i$ is non-zero is using the leave-one-out approach [38] to test the externally studentized residual

*Corresponding author.

$$t_i = \frac{\boldsymbol{y}_i - \boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}_{-i}}{\hat{\sigma}_{-i}\left(1 + \boldsymbol{x}_i^\top \left(\boldsymbol{X}_{-i}^\top \boldsymbol{X}_{-i}\right)^{-1} \boldsymbol{x}_i\right)^{1/2}}, \qquad (2)$$

where $\hat{\sigma}$ is the scale estimate and the subscript $-i$ indicates estimates based on the $n-1$ observations, leaving out the i-th data where we are testing. Equivalently, the linear regression model can be re-formulated into explicitly representing the residual by the mean-shift parameter $\boldsymbol{\gamma}$ as in [39],

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\gamma} + \boldsymbol{\varepsilon}, \quad \varepsilon_{i,j} \sim \mathcal{N}(0, \sigma^2), \qquad (3)$$

where we have the feature $\boldsymbol{X} \in \mathbb{R}^{n\times p}$, and label $\boldsymbol{Y} \in \mathbb{R}^{n\times c}$ paired and stacked by rows; and each row of $\boldsymbol{\gamma} \in \mathbb{R}^{n\times c}$, $\boldsymbol{\gamma}_i$, represents the predict residual of the corresponding data. This formulation has been widely studied in different research topics, including economics [4, 18, 32, 33], robust regression [8, 39], statistical ranking [9], face recognition [56], semi-supervised few-shot learning [54, 55], and Bayesian preference learning [43], to name a few. The focused formulation is different depending on the specific research tasks. For example, for the robust regression problem, the target is to get a robust estimate $\hat{\boldsymbol{\beta}}$ against the influence of $\boldsymbol{\gamma}$. Here for solving the problem of learning with noisy labels, we instead aim to *amplify* the impact of $\boldsymbol{\gamma}$ such that non-zero values can represent the noisy label that existed in the training set.

To this end, from the statistical perspective, this paper starts from Eq. (3) to build up a sample selection framework, dubbed *Scalable Penalized Regression* (SPR), which has theoretical guarantees of consistently identifying noisy data, and thus can efficiently learn with noisy labels. Naturally, we expect $\boldsymbol{\gamma}$ in Eq. (3) to be sparse and only a small number of $\boldsymbol{\gamma}_i$ are non-zeros, indicating that those data are noisy or outlying. Thus a sparse penalty is utilized on $\boldsymbol{\gamma}_i$ to encourage that the non-zero solution is restricted in a small portion. We thus optimize the induced penalized regression problem to solve $\boldsymbol{\gamma}$ and identify the instances with non-zero $\boldsymbol{\gamma}_i$ as noisy data. Theoretically, in terms of the model selection consistency theory [51, 66], there is some nice statistical property and theoretical insight in our SPR framework, as we can guarantee that, by meeting certain conditions, our SPR should at least in principle, successfully identify all the noisy data.

To incorporate Eq. (3) into the end-to-end training pipeline of deep architecture, the simplest way is to solve Eq. (3) for each training mini-batch to detect and remove noisy data. However, when we train large model with small batch size, the information of current mini-batch may not be identifiable enough to distinguish true pattern from noise. On the other hand, use SPR on the whole training data after training an epoch leads to an unacceptable computation cost due to the quadratically increased complexity of solving Eq. (3) with the training data. To design a proper optimization environment for solving Eq. (3) that is data-efficient and identifiable, we utilize the whole training set and propose a split algorithm to divide it into small pieces that are class balance with proper data size such that the noisy pattern is *identifiable* and can be solved *efficiently* in parallel, making SPR scalable to large datasets.

Inspired by [69], to further encourage the linear relation between features and labels, we propose using a sparse penalty on the fully-connected output before it is soft-maxed. Moreover, we utilize SPR to train the network in a semi-supervised manner using CutMix [62], regarding the detected noisy data as unlabeled data to fully utilize the feature information. We conduct extensive experiments to validate the effectiveness of our framework on several benchmark datasets and real-world noisy datasets.

Contributions. Our contributions are as follows:

- We present a statistical approach, SPR, to identify noisy data under a general scenario with theoretical guarantees.
- A split algorithm is proposed to make SPR scalable to large datasets.
- A sparse penalty is proposed to encourage the linear relation, and a full training framework that combines SPR with semi-supervised methods is designed.
- Experiments on benchmark datasets and real-world noisy datasets validate the effectiveness of SPR.

2. Related Work

The target of Learning with Noisy Labels (LNL) is to train a more robust model from the noisy dataset. We can roughly categorize LNL algorithms into two groups: robust algorithm and noise detection. Robust algorithm does not focus on specific noisy data, but designs specific modules to ensure that networks can be well-trained even from the noisy datasets. Methods following this direction includes constructing robust network [6, 12, 13, 59], robust loss function [11, 27, 53, 65, 68, 69], robust regularization [31, 47, 58] against noisy labels.

Noise detection method aims to identify the noisy data and design specific strategies to deal with the noisy data, including down-weighting the importance in the loss function for the network training [48], re-labeling them to get correct labels [46], or regarding them as unlabeled data in the semi-supervised manner [23], etc.

For the noise detection algorithm, noisy data are identified by some irregular patterns, including large error [40], gradient directions [37], disagreement within multiple networks [61], inconsistency along the training path [67] and some spatial properties in the training data [22, 52, 57].

Some algorithms [37, 50] rely on the existence of an extra clean set to detect noisy data.

After detecting the clean data, the simplest strategy is to train the network using the clean data only or re-weight the data [35] to eliminate the noise. Some algorithms [2, 23] regard the detected noisy data as unlabeled data to fully exploit the distribution support of the training set in the semi-supervised learning manner. There are also some studies of designing label-correction module [25,46,49,50, 59,60] to further pseudo-labeling the noisy data to train the network. Few of those approaches are designed from the statistical perspective with non-asymptotic guarantees. In this paper, we propose to use SPR to identify the noisy data under general scenarios with statistical guarantees.

3. Methodology

Problem Formulation. We are given a dataset of image-label pairs $\{(\boldsymbol{I}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{I}_i \in \mathcal{I} \subseteq \mathbb{R}^m, y_i \in \mathcal{C} \subseteq \mathbb{R}, |\mathcal{C}| = c$, with the one-hot encoding of y_i as $\boldsymbol{y}_i$. We assume that for each instance i, y_i is corrupted from the ground-truth category $y_i^\star$, where the ground-truth and corruption process is unknown. Our goal is predicting the ground-truth label $y^\star \in \mathcal{C}$ for any $\boldsymbol{I} \in \mathcal{I}$, by a neural network composed of a feature extractor $f(\cdot)$ and a classifier $g(\cdot)$. Typically the network first encodes the image $\boldsymbol{I}_i$ as a feature vector $\boldsymbol{x}_i = f(\boldsymbol{I}_i)$, and return the soft-max probability $\hat{\boldsymbol{y}}_i = g(\boldsymbol{x}_i)$.

We present our framework – *Scalable Penalized Regression* (SPR), designed as a sample selection component to the training pipeline of neural networks. As mentioned in the introduction, SPR is motivated by the leave-one-out approach [38] on the t-test of prediction residuals to identify and remove the noisy data for the network to train, by solving the mean-shift parameter in a sparse linear regression model (Eq. (3)). Specifically, we use a sparse linear regression model to fit the feature-label pairs $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_i^n$ received from the current training time, and solve the corresponding mean-shift parameter γ in

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} \frac{1}{2} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_{\mathrm{F}}^2 + \sum_{i=1}^n P(\gamma_i; \lambda_i), \qquad (4)$$

where $P(\cdot;\cdot)$ is a sparse regularization on $\boldsymbol{\gamma}$ with coefficient λ_i on row $\boldsymbol{\gamma}_i$ to ensure that non-zero $\boldsymbol{\gamma}_i$ are sparse, whose corresponding instance are identified as the noisy data. We denote $\boldsymbol{A}_i, \boldsymbol{A}_{\cdot,j}, \|\boldsymbol{A}\|_{\mathrm{F}}^2 := \sum_{i,j} A_{i,j}^2$ as the i-th row, j-th column and the square of Frobenius norm, respectively.

3.1. Preliminary: Penalized Regression in Statistics

The penalized regression problem (Eq. (4)) is widely studied in statistics, where the standard solving algorithm is an alternating optimization pipeline: for fixed $\boldsymbol{\gamma}$, the global optimal solution of $\boldsymbol{\beta}$ is the Ordinary Least Square (OLS) estimate of the linear regression problem on $(\boldsymbol{X}, \boldsymbol{Y} - \boldsymbol{\gamma})$; while for fixed $\boldsymbol{\beta}$, the problem is separable in each row of $\boldsymbol{\gamma}$, which can be solved by soft-thresholding. Further, it is shown in [1, 10] that the penalized regression problem enjoys the same optimal solution of Huber's M-estimate [16], which minimizes

$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^n \rho\left(\frac{\boldsymbol{y}_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}}{\sigma}; \lambda\right) + \frac{1}{2} cn\sigma, \qquad (5)$$

for fixed constants $\sigma > 0, c \geq 0, \lambda > 0$, where $\rho(t;\lambda) = t^2/2$ when $|t| \leq \lambda$ and $\rho(t;\lambda) = \lambda|t| - \lambda^2/2$ otherwise. The general formulation of the penalty P can be defined with a three-step construction algorithm introduced in [39]. In our experiments, we use the ℓ_1 norm as the penalty.

3.2. Penalized Regression for LNL

In this paper, we regard $\boldsymbol{\gamma}$ as the indicator of noisy data, with larger $\|\boldsymbol{\gamma}_i\|$ means more corruption the instance i is suffered. We denote $O := \{i : \|\boldsymbol{\gamma}_i\| \neq 0\}$ as the noisy sample set. To estimate O, we only need to solve $\boldsymbol{\gamma}$ with no need to estimate $\boldsymbol{\beta}$. Thus to simplify the optimization, we substitute the OLS estimate for $\boldsymbol{\beta}$ with γ fixed into Eq. (4). To ensure that $\hat{\boldsymbol{\beta}}$ is identifiable, we apply PCA on $\boldsymbol{X}$ to make $p \ll n$ so that the $\boldsymbol{X}$ has full-column rank. Denote $\tilde{\boldsymbol{X}} = \boldsymbol{I} - \boldsymbol{X}\left(\boldsymbol{X}^\top \boldsymbol{X}\right)^\dagger \boldsymbol{X}^\top, \tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{X}}\boldsymbol{Y}$, the Eq. (4) is transformed into

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \frac{1}{2} \left\|\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma}\right\|_{\mathrm{F}}^2 + \sum_{i=1}^n P(\gamma_i; \lambda_i), \qquad (6)$$

which is a standard sparse linear regression for $\boldsymbol{\gamma}$. Note that in practice we can hardly choose a proper λ that works well in all scenarios. Furthermore, from the equivalence between the penalized regression problem and Huber's M-estimate, the solution of $\boldsymbol{\gamma}$ is returned with soft-thresholding. Thus it is not worth to find the precise solution of a single $\boldsymbol{\gamma}$. Instead, we use a block-wise descent algorithm [41] to solve $\boldsymbol{\gamma}$ with a list of λs and generate the solution path. As λ changes from ∞ to 0, the influence of sparse penalty decreases, and $\boldsymbol{\gamma}_i$ are gradually solved with non-zero values, in other words, selected by the model. Since earlier selected instance is more possible to be noisy, we rank all samples as the descendent order of their selecting time defined as:

$$C_i = \sup\{\lambda : \boldsymbol{\gamma}_i(\lambda) \neq 0\}. \qquad (7)$$

A large C_i means that the $\boldsymbol{\gamma}_i$ is earlier selected. Then the top samples are identified as noisy data.

3.3. Scalable Penalized Regression

The computation cost of Eq. (6) is $O(n^2 c)$, which increases in quadratic with the growth of the training sample, making it not scalable to large datasets. Note that we are finding data that are more noisy than others from Eq. (7); thus we may generate the environment that the noisy instances are easier to be identified and with less computation cost. To this end, we propose to split the total training dataset into many pieces, each of which contains a small portion of training categories with a small number of training data. With the splitting strategy, SPR can run on several pieces in parallel and significantly reduce the running time.

Recall that the principle of selecting the group of categories is to reduce the optimizing difficulty and generate an easier environment for finding noisy data. Our motivation is that categories with less similarity are helpful to identify the noisy data (based on the noisy set recovery theory which we will introduce later), where the similarity is defined as

$$s(i,j) = \boldsymbol{p}_i^\top \boldsymbol{p}_j, \tag{8}$$

for class i, j where $\boldsymbol{p}$ represents the class prototype. Specifically, we take the clean features $\boldsymbol{x}_i$ of each class extracted by the network along the training iteration, and average them to get the class prototype $\boldsymbol{p}_c$ after the current training epoch ends, as

$$\boldsymbol{p}_c = \frac{\sum_{i=1, y_i=c, i\notin O}^{n} \boldsymbol{x}_i}{\sum_{i=1, y_i=c, i\notin O}^{n} 1}, \tag{9}$$

Then the most dissimilar classes are grouped together. In our experiments, we design one group with 10 classes.

For the instances in each group, we split the training data of each class in a balanced way such that each group contains the same number of instances for each class. The number is determined to ensure that the clean pattern remains the majority in the group, such that optimization can be done easily. In practice, we select 10 training data of each class to construct the group. When there is an imbalance between different classes, we use over-sampling strategy to sample the instance of class with less training data multiple times to ensure that each training instance is selected once in some split group. The detection process is shown in Algorithm 1.

3.4. Learning with Detected Noisy Data

Supervised training manner. After estimating the noisy set O, the simplest strategy is to remove them and train the network with the remaining clean data. We show that this strategy will lead to an improvement in testing accuracy. Note that we assume in Eq. (3) that the one-hot encoded label is linearly related to the feature $\boldsymbol{X}$; however, in practice, the prediction is obtained via the soft-max function on the $\boldsymbol{X}\boldsymbol{W}_{\mathrm{fc}}$, where $\boldsymbol{W}_{\mathrm{fc}}$ is the weight of the final fully-connected layer (we ignore the bias term for simplicity).

Algorithm 1: Scalable Regularized Regression

```
Input: Feature matrix $\boldsymbol{X}$, label matrix $\boldsymbol{Y}$, noisy set $O$.
Calculate prototypes $\boldsymbol{P}$ of each class using Eq. (9);
Divide classes into most diverged groups based on the
  similarity within classes using Eq. (8);
Split data of each class in the same group into pieces
  $\{(\boldsymbol{X}_{(i)}, \boldsymbol{Y}_{(i)})\}$;
for each piece of $(\boldsymbol{X}_{(i)}, \boldsymbol{Y}_{(i)})$ in parallel do
    Solve $\boldsymbol{\gamma}_{(i)}$ using Eq. (6);
    Select the noisy subset $O_{(i)}$ using Eq. (7);
end
Group all the $O_{(i)}$ together and return the result $O$.
```

To reduce this gap, inspired by [69], we append a ℓ_q ($q < 1$) penalty on the cross entropy loss, which encourages the linear relationship between $\boldsymbol{X}$ and one-hot encoded vector $\boldsymbol{Y}$:

$$\mathcal{L}\left(\boldsymbol{x}_i, \boldsymbol{y}_i\right) = 1_{i\notin O}(\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{x}_i, \boldsymbol{y}_i\right) + \lambda \|\boldsymbol{x}_i^\top W_{\mathrm{fc}}\|_q), \tag{10}$$

where $q < 1$, $\mathcal{L}_{\mathrm{CE}}$ denotes the cross-entropy loss, and $1_{i\notin O}$ denotes the indicator function such that the loss is only calculated on the clean data. Note that the $\|\boldsymbol{x}^\top W_{\mathrm{fc}}\|_q$ enforces the $\boldsymbol{x}^\top W_{\mathrm{fc}}$ to approximately be one-hot encoded vector as long as q is small enough. In this training manner, SPR can be regarded as a robust loss function algorithm since we do not modify the training pipeline except the modification of the loss function.

Semi-supervised training manner. We can further exploit the support of noisy data by incorporating SPR with semi-supervised algorithms. In this paper, we interpolate part of images between clean data and noisy data as in [62],

$$\tilde{\boldsymbol{x}} = \boldsymbol{M} \odot \boldsymbol{x}_{\text{clean}} + (1 - \boldsymbol{M}) \odot \boldsymbol{x}_{\text{noisy}} \tag{11a}$$

$$\tilde{\boldsymbol{y}} = \lambda \boldsymbol{y}_{\text{clean}} + (1 - \lambda) \boldsymbol{y}_{\text{noisy}} \tag{11b}$$

where $\boldsymbol{M} \in \{0,1\}^{W\times H}$ is a binary mask, $\odot$ is element-wise multiplication, and the clean and noisy data are identified by SPR. Then we train the network using the interpolated data using

$$\mathcal{L}\left(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}\right) = \mathcal{L}_{\mathrm{CE}}\left(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}\right). \tag{12}$$

Since $\tilde{\boldsymbol{y}}$ is interpolated, it is no longer a one-hot vector, and thus is not sparse. Hence we do not use the sparse penalty when we train the network using the interpolated data. Note that SPR is done using the original data without interpolation, hence the linear relationship still holds. In practice, the above two training method is randomly selected in each mini-batch with the predefined probability. The full algorithm is shown in Algorithm 2.

Algorithm 2: Training algorithm

```
Initialize: Noisy dataset $\{(\boldsymbol{I}_i, \boldsymbol{y}_i)\}_{i=1}^n$, feature matrix
  $\boldsymbol{X}$, noisy label matrix $\boldsymbol{Y}$, noisy set $O = \phi$, CutMix
  probability $p$.
for ep from 0 to total epochs do
    for each mini-batch do
        Sample $r$ from $U(0,1)$;
        if $r > p$ then
            Train the network using Eq. (10).
        else
            Train the network using Eq. (12).
        end
        Update $\boldsymbol{X}$ visited in the current mini-batch;
    end
    Run SPR (Algorithm 1) on $(\boldsymbol{X}, \boldsymbol{Y})$ and update
      noisy set $O$;
end
```

3.5. Noisy Set Recovery of SPR

In this part, we provide the result that the Eq. (4) can recover the oracle support set O. For simplicity, we use the ℓ_1 norm as the penalty. In the above we have re-formulate Eq. (4) as Eq. (6), which is a standard multi-response regression problem. Here we further vectorize the problem such that it shares the standard formulation of LASSO. Then we can use the well-studied model selection consistency result [51, 66] to support our conclusion. Specifically, we vectorize $\boldsymbol{Y}, \gamma$ in Eq. (6) as $\vec{\boldsymbol{y}}, \vec{\gamma}$ and the Eq. (6) turns to

$$\underset{\vec{\gamma}}{\operatorname{argmin}} \frac{1}{2} \left\| \vec{\boldsymbol{y}} - \mathring{\boldsymbol{X}} \vec{\gamma} \right\|_2^2 + \lambda \left\| \vec{\gamma} \right\|_1, \tag{13}$$

where $\mathring{\boldsymbol{X}} = I_c \otimes \tilde{\boldsymbol{X}}$ with $\otimes$ denoting the Kronecker product operator. Denote $S := \operatorname{supp}(\vec{\gamma}^*)$, then it is sufficient for the recovery of noisy set O to recover S. We further denote $\mathring{\boldsymbol{X}}_S (\mathring{\boldsymbol{X}}_{S^c})$ as the column vectors of $\mathring{\boldsymbol{X}}$ whose indexes are in $S(S^c)$ and $\mu_{\mathring{\boldsymbol{X}}} = \max_{i \in S^c} \|\mathring{\boldsymbol{X}}\|_2^2$. Then we have

Theorem 1 (Noisy set recovery). *Assume that:*
C1, Restricted eigenvalue: $\lambda_{\min}(\mathring{\boldsymbol{X}}_S^\top \mathring{\boldsymbol{X}}_S) = C_{\min} > 0$;
C2, Irrepresentability: *there exists a* $\eta \in (0,1]$, *such that* $\|\mathring{\boldsymbol{X}}_{S^c}^\top \mathring{\boldsymbol{X}}_S (\mathring{\boldsymbol{X}}_S^\top \mathring{\boldsymbol{X}}_S)^{-1}\|_\infty \leq 1 - \eta$;
C3, Large error: $\vec{\gamma}^*_{\min} := \min_{i \in S} |\vec{\gamma}^*_i| > h(\lambda, \eta, \mathring{\boldsymbol{X}}, \vec{\gamma}^*)$; *where* $\|\boldsymbol{A}\|_\infty := \max_i \sum_j |A_{i,j}|$, *and* $h(\lambda, \eta, \mathring{\boldsymbol{X}}, \vec{\gamma}^*) = \lambda\eta / \sqrt{C_{\min} \mu_{\mathring{\boldsymbol{X}}}} + \lambda \|(\mathring{\boldsymbol{X}}_S^\top \mathring{\boldsymbol{X}}_S)^{-1} \operatorname{sign}(\vec{\gamma}^*_S)\|_\infty$.
Let $\lambda \geq \frac{2\sigma\sqrt{\mu_{\mathring{\boldsymbol{X}}}}}{\eta} \sqrt{\log cn}$. *Then with probability greater than* $1 - 2(cn)^{-1}$, *model Eq.* (13) *has a unique solution* $\hat{\vec{\gamma}}$ *such that: 1) If C1 and C2 hold,* $\hat{O} \subseteq O$*;2) If C1, C2 and C3 hold,* $\hat{O} = O$.

Note that The Theorem 1 is extended from the model selection consistency in [51], which only provides the conclusion that $\hat{S} \subseteq S$ and $\hat{S} = S$, respectively. Here we show that $\hat{S} \subseteq S$ leads to $\hat{O} \subseteq O$, and of course $\hat{S} = S$ leads to $\hat{O} = O$ in our case. For instance i, $i \in O^c$ only when $\gamma_{i,j} = 0$ for all j, then all the vectorized indexes are in S^c. When $\hat{S} \subseteq S$, all vectorized indexes of instance i are in $\hat{S}^c$, which means $i \in \hat{O}^c$ and leads to $\hat{O} \subseteq O$.

C1 is necessary to get a unique solution, and in our case is mostly satisfied with the nature assumption that the clean data is the majority in the training data. If C2 holds, the estimated noisy data is the subset of truly noisy data. This condition is the key to ensuring the success of SPR, which requires divergence between clean and noisy data such that we cannot represent clean data by noisy data. If C3 further holds, the estimated noisy data is exactly all the truly noisy data. C3 requires the error measured by γ_i is large enough to be identified from random noise. If the conditions fail, SPR will fail in a non-vanishing probability, not deterministic.

4. Experiments

Datasets. We validate the effectiveness of SPR on synthetic noisy datasets MNIST [21] and CIFAR10 [20], and real-world noisy datasets ANIMAL10 [44] and WebVision [24]. We consider two types of noisy labels for MNIST and CIFAR10: (i) Symmetric noise: Every class is corrupted uniformly with all other labels; (ii) Asymmetric noise: Labels are corrupted by similar (in pattern) classes. The ANIMAL10 is published with mislabeling (the ratio is 8%) and the corruption process and noise type in ANIMAL10 are unknown. WebVision has 2.4 million images collected from the internet with the same category list with ImageNet ILSVRC12. Thus, the ANIMAL10 and WebVision datasets can be regarded as a real-world challenge.

Backbones. For MNIST, we use two convolutional layers followed by two fully-connected layers, denoted as *C2F2*. For CIFAR10, a ResNet-18 [15] network is utilized. For ANIMAL10 we use VGG19-BN [42] as our backbone. And for WebVision we use Inception-ResNet [45] to extract features.

Hyperparameter setting. We use SGD to train all the networks with momentum 0.9 and a cosine learning rate decay strategy. The initial learning rate is set as 0.1 for ANIMAL10 and 0.01 for others. The weight decay is set as 1e-3,1e-4, 1e-3, 5e-4 for MNIST, CIFAR10, ANIMAL10, and WebVision, respectively. We use a batch size of 128 for all experiments. We use random crop and random horizontal flip as augmentation strategies for CIFAR10, ANIMAL10, and WebVision. The network is trained for 50 epochs for MNIST, 180 epochs for CIFAR10, 160 epochs for ANIMAL10, and 300 epochs for WebVision. We use $q = 0.2$ in Eq. (10) with coefficient λ of the sparse penalty initialized as 1.2 and is increased by multiplying 1.2 for

Dataset	Method	Sym. Noise Rate				Asy. Noise Rate		
		0.2	0.4	0.6	0.8	0.2	0.3	0.4
MNIST (C2F2)	CE	91.6	74.0	49.4	22.7	94.6	88.8	82.3
	FL	91.7	74.5	50.4	22.7	94.3	89.1	82.1
	GCE	98.9	97.2	81.5	34.0	96.7	89.1	81.5
	SCE	98.9	97.4	88.8	48.8	98.0	93.7	85.4
	NLNL	98.3	97.8	96.2	86.3	98.4	97.5	95.8
	APL	99.1	98.4	95.7	73.0	98.9	96.9	91.5
	SR	99.2	99.2	98.9	98.0	99.3	99.2	99.2
	SPR	**99.3**	**99.2**	**99.2**	**98.7**	**99.3**	**99.2**	**99.2**
CIFAR-10 (ResNet-18)	Standard	85.7	81.8	73.7	42.0	88.0	86.4	84.9
	Forgetting	86.0	82.1	75.5	41.3	89.5	88.2	85.0
	Bootstrap	86.4	82.5	75.2	42.1	88.8	87.5	85.1
	Forward	85.7	81.0	73.3	31.6	88.5	87.3	85.3
	Decoupling	87.4	83.3	73.8	36.0	89.3	88.1	85.1
	MentorNet	88.1	81.4	70.4	31.3	86.3	84.8	78.7
	Co-teaching	89.2	86.4	79.0	22.9	90.0	88.2	78.4
	Co-teaching+	89.8	86.1	74.0	17.9	89.4	87.1	71.3
	IterNLD	87.9	83.7	74.1	38.0	89.3	88.8	85.0
	RoG	89.2	83.5	77.9	29.1	89.6	88.4	86.2
	PENCIL	88.2	86.6	74.3	45.3	90.2	88.3	84.5
	GCE	88.7	84.7	76.1	41.7	88.1	86.0	81.4
	SCE	89.2	85.3	78.0	44.4	88.7	86.3	81.4
	TopoFilter	90.2	87.2	80.5	45.7	90.5	89.7	87.9
	SPR	**93.2**	**91.0**	**82.7**	**64.1**	**92.8**	**91.3**	**89.0**

Table 1. Test accuracies on several benchmark datasets with different settings. The best result is **boldfaced**. Results of competitors on MNIST are reported in [69], and on CIFAR10 are reported in [57].

MNIST, and 1.02 for others. In CIFAR10 with noise rate 0.8, we do not increase the λ. We simply select half of the training data as noisy data in all of our experiments.

4.1. Evaluation on Synthetic Label Noise

Competitors. In this part, we first use SPR with only using the supervised training manner on MNIST to compare with robust loss function methods. Then we use the full SPR model on CIFAR-10 to compare with sample selection algorithms and other popular algorithms. We use cross-entropy loss (CE) as baseline algorithm for two datasets. For MNIST, we also compare with competitors including an effective loss function Focal Loss (FL) [26], some refined algorithms for CE loss like GCE [65] and SCE [53], NLNL [19] which utilizes complementary labels against the noise, APL [28] which combines robust active and passive loss to train the network. SR [69] which utilizes the sparse regularization combined with the feature normalization and temperature scaling method to train the network. For CIFAR-10, we compare SPR with algorithms include Forgetting [3] with train the network using dropout strategy, Bootstrap [36] which train with bootstrapping, Forward Correction [35] which corrects the loss function to get a robust model, Decoupling [30] which uses a meta update strategy to decouple the update time and update method, MentorNet [17] which uses a teacher network to help train the network, Co-teaching [14] which uses two networks to teach each other, Co-teaching+ [61] which further uses an update by disagreement strategy to improve Co-teaching, IterNLD [52] which uses an iterative update strategy, RoG [22] which uses generated classifiers, PENCIL [60] which uses a probabilistic noise correction strategy, GCE [65] and SCE [53] which are extensions of standard cross-entropy loss function, and TopoFilter [57] which uses feature representation to detect noisy data. For each dataset, all the experiments are running with the same backbone to make a fair comparison.

As in Tab. 1, SPR enjoys a high performance compared with other robust loss function algorithm without using noisy data on MNIST, and shows a high superiority to many competitors on CIFAR-10, validating the effectiveness of SPR on different noise scenarios.

4.2. Evaluation on Real-World Noisy Datasets

In this part, we compare SPR with other methods in real-world noisy datasets including ANIMAL10 and WebVision.

For WebVision we use CutMix probability of 1.0.

Competitors. For ANIMAL10, we compare with the baseline of directly training with cross-entropy loss (CE), as well as previous works including Nested(ND), CE + Dropout (CED), SELFIE [44], PLC [64], and NestedCoTeaching (NCT) [7]. For WebVision, we compare with directly training with cross-entropy loss (CE), as well as Decoupling [30], D2L [29], MentorNet [17], Co-teaching [14], Iterative-CV [5], and DivideMix [23].

The results of real-world datasets are shown in Tab. 2, where the results of CE and SELFIE on ANIMAL is reported in [44], the results of ND, CED, and NCT is reported in [7], while the result of PLC is reported in their paper. The results of competitors on WebVision are reported in [23] and the result of CE is reported in [69]. Our algorithm SPR enjoys superior performance to all the competitors, showing the ability of handling real-world challenges.

ANIMAL10		WebVision	
Model	Accuracy	Model	Accuracy
CE	79.4	CE	66.96
Nested	81.3	Decoupling	62.54
CED	81.3	MentorNet	63.00
SELFIE	81.8	Co-teaching	63.58
PLC	83.4	Iterative-CV	65.24
NCT	84.1	DivideMix	77.32
SPR	**86.8**	SPR	**78.12**

Table 2. Results on real-world datasets. The best are in **bold**.

4.3. More Analysis of SPR

Precision of noisy detection. Besides accuracy, another metric to test the capacity of a sample selection algorithm is *label precision*: the ratio of true clean labels in the detected clean instances. In this part, we check the label precision of SPR to show the sample selection effectiveness. We conduct our experiments on the symmetric noise rate of 0.4 and 0.8, as well as asymmetric noise rate of 0.4. Results are shown in Fig. 1. SPR enjoys a monotonically increasing label precision in the symmetric noise setting, leading to a better training environment than the standard noisy dataset. When the training process ends, almost all the selected training data is guaranteed to be clean data (93.90% in the symmetric-40% setting). In the symmetric-80% setting, due to the strategy of selecting half of the training data, the upper bound of the precision is 40%, as illustrated. In this high noise rate scenario, SPR can still achieve the precision of 30.34%, which means that 76.24% of the clean training instances are detected by our algorithm. Note that in the asymmetric-40%, the label precision is first increased then decreased, and ends with 93.50%. Though it is still high, the accuracy drops with the label precision, suggesting that an early stopping strategy is needed in the asymmetric noisy setting. We leave it as a future work to provide a fine-grained framework for different noise scenarios.

Model	Accuracy
CE	65.5
CE + SPR	80.4
CE + ℓ_q	71.6
CE + CutMix	87.0
CE + SPR + ℓ_q	88.5
CE + SPR + CutMix	89.2
Full	91.0

Table 3. Accuracy of using different modules in SPR.

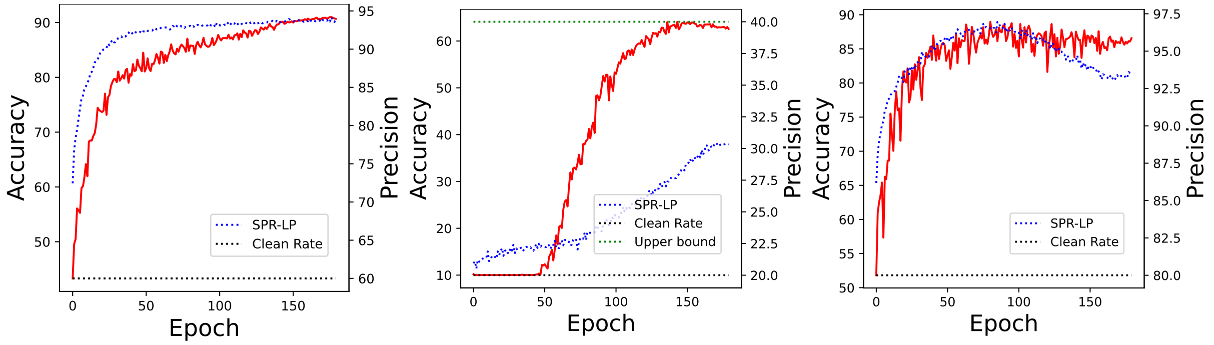
Ablation study of modules in SPR. To verify the effectiveness of each module in our framework, we conduct an ablation study on CIFAR10 with 40% symmetric noise rate. Specifically, the "CE" denotes vanilla cross entropy method; the "CE + SPR" means the cross-entropy loss only on the clean data detected by SPR; the "CE + ℓ_q" means the Eq. (10) for all training data; the "CE + CutMix" means using CutMix strategy for all the training data; other variants are defined similarly based on utilized components, and the "Full" denotes our SPR method with all components. As shown in Tab. 3, simply using our framework to detect noisy data will lead to better performance compared with the standard CE loss. And the full model enjoys the best performance. We further visualize the learned representation of using SPR compared with standard cross-entropy method in Fig. 2. SPR will learn a better discriminative representations.

Model	Training Time
SPR w/o split algorithm	about 6h
SPR w/ split algorithm	54s

Table 4. Training time for one epoch on CIFAR-10.

Influence of Split algorithm. In our framework, we propose a split algorithm to divide the whole training set into small pieces to run SPR in parallel. In this part, we compare the running time between using the split algorithm and not using it. Results are shown in Tab. 4. When we do not use the split algorithm, the training time for each epoch will cost an unacceptable time, making it impossible to train on large datasets. Hence we propose the split algorithm to reduce the training time.

Influence of select ratio. In our experiments, we simply select half of the training data to train the network. It is desirable to investigate how does the ratio of detected noisy data influence the training process. We conduct experiments of SPR on CIFAR10 with a symmetric noise rate of 0.8. To

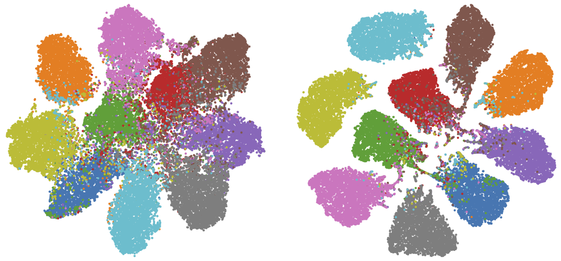


(a) Symmetric-40%

(b) Symmetric-80%

(c) Asymmetric-40%

Figure 1. Accuracy and Label precision of SPR under different noise scenarios. The red line is the accuracy of SPR, while the dotted line is the label precision.



(a) CE

(b) SPR

Figure 2. Visualization of learned representations.

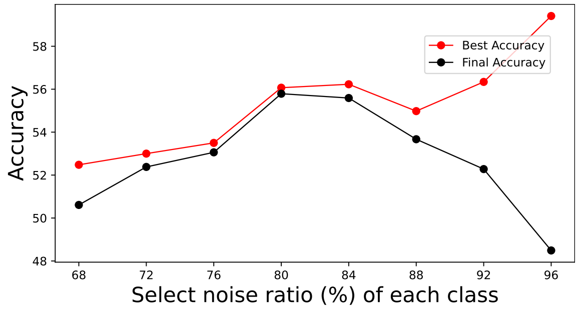


Figure 3. Best and final accuracies of SPR running with selecting different ratio of training data.

avoid the influence of semi-supervised training pipeline, we only use the supervised training manner in this part. It can be found that the best selection ratio is near the noise ratio in the training set. Hence a better selection strategy may be designed based on the estimation of the noise ratio in the training set. We leave it as a future work since in this paper we mainly propose the sample selection framework.

Influence of ℓ_q. In this part, we investigate the influence of ℓ_q norm in our framework. We run with a sequence of q from 0.05 to 1, as illustrated in Fig. 4. In general, a smaller q encourages the linear relation as expected by our framework, while too small q will damage the representation capacity of the network. Thus, a convex accuracy curve exists, suggesting a choice of $q = 0.2$ to be the best. Hence in our experiments we use $q = 0.2$.

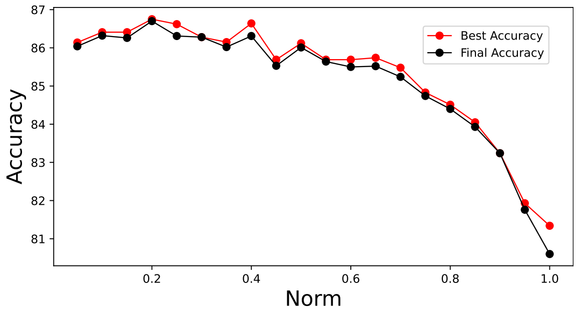


Figure 4. Accuracies of SPR with different ℓ_q.

Limitations of SPR. The major limitation of SPR is that it requires the almost necessary irrepresentability condition to identify the noise set. When this condition is not hold for the problem, SPR will end with a non-vanishing probability to identify at least one clean data as noisy data. Further, the recovery theory are based on the Gaussian noise assumption, which may not hold for special problems.

5. Conclusion

This paper proposes a statistical sample selection framework – Scalable Penalized Regression (SPR) to identify noisy data with theoretical guarantees. Specifically, we propose an equivalent leave-one-out t-test approach as a penalized linear model, in which non-zero mean-shift parameters can be induced as an indicator for noisy data. We provide theoretical conditions to guarantee the identifiability of SPR to recover the oracle noisy set. Experiments on several synthetic and real-world datasets show the effectiveness of our method.

Social Impact. Our SPR will have positive impact to social, as it enables to directly identify noisy data with theoretical grounding to help train the network.

Acknowledgement. This work was supported in part by NSFC Project (62176061), and Shanghai Municipal Science and Technology Major Project (2018SHZDZX01).

References

[1] Anestis Antoniadis. Wavelet methods in statistics: Some recent developments and their applications. *Statistics Surveys*, 2007. 3

[2] Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *ICML*, 2019. 3

[3] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *ICML*, 2017. 6

[4] Debabrata Basu. On the elimination of nuisance parameters. In *Selected Works of Debabrata Basu*. 2011. 2

[5] Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and utilizing deep neural networks trained with noisy labels. In *ICML*, 2019. 1, 7

[6] Xinlei Chen and Abhinav Gupta. Webly supervised learning of convolutional networks. In *ICCV*, 2015. 1, 2

[7] Yingyi Chen, Xi Shen, Shell Xu Hu, and Johan AK Suykens. Boosting co-teaching with compression regularization for label noise. In *CVPR workshop*, 2021. 7

[8] Jianqing Fan, Runlong Tang, and Xiaofeng Shi. Partial consistency with sparse incidental parameters. *Statistica Sinica*, 2018. 2

[9] Yanwei Fu, Timothy M Hospedales, Tao Xiang, Jiechao Xiong, Shaogang Gong, Yizhou Wang, and Yuan Yao. Robust subjective visual property prediction from crowdsourced pairwise labels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 2

[10] Irène Gannaz. Robust estimation and wavelet thresholding in partially linear models. *Statistics and Computing*, 2007. 3

[11] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. In *AAAI*, 2017. 1, 2

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

[13] Bo Han, Jiangchao Yao, Gang Niu, Mingyuan Zhou, Ivor W Tsang, Ya Zhang, and Masashi Sugiyama. Masking: a new perspective of noisy supervision. In *NeurIPS*, 2018. 1, 2

[14] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 2018. 1, 6, 7

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

[16] Peter J Huber. *Robust statistics*. 2004. 3

[17] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2018. 1, 6, 7

[18] Jack Kiefer and Jacob Wolfowitz. Consistency of the maximum likelihood estimator in the presence of infinitely many incidental parameters. *The Annals of Mathematical Statistics*, 1956. 2

[19] Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. Nlnl: Negative learning for noisy labels. In *ICCV*, 2019. 6

[20] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 5

[22] Kimin Lee, Sukmin Yun, Kibok Lee, Honglak Lee, Bo Li, and Jinwoo Shin. Robust inference via generative classifiers for handling noisy labels. In *ICML*, 2019. 2, 6

[23] Junnan Li, Richard Socher, and Steven C.H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2020. 2, 3, 7

[24] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017. 5

[25] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *ICCV*, 2017. 3

[26] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 6

[27] Yueming Lyu and Ivor W Tsang. Curriculum loss: Robust learning and generalization against label corruption. In *ICLR*, 2020. 1, 2

[28] Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah Erfani, and James Bailey. Normalized loss functions for deep learning with noisy labels. In *ICML*, 2020. 6

[29] Xingjun Ma, Yisen Wang, Michael E Houle, Shuo Zhou, Sarah Erfani, Shutao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. In *ICML*, 2018. 7

[30] Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". In *NeurIPS*, 2017. 6, 7

[31] Aditya Krishna Menon, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Can gradient clipping mitigate label noise? In *ICLR*, 2020. 2

[32] Marcelo Moreira. A maximum likelihood method for the incidental parameter problem. Technical report, 2008. 2

[33] Jerzy Neyman and Elizabeth L Scott. Consistent estimates based on partially consistent observations. *Econometrica: Journal of the Econometric Society*, 1948. 2

[34] Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. Self: Learning to filter noisy labels with self-ensembling. In *ICLR*, 2020. 1

[35] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, 2017. 3, 6

[36] Scott E Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR (Workshop)*, 2015. 6

[37] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, 2018. 2, 3

[38] Weisberg Sanford. Applied linear regression. *John Wiley & Sons*, 1985. 1, 3

[39] Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. *Journal of the American Statistical Association*, 2011. 2, 3

[40] Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In *ICML*, 2019. 1, 2

[41] Noah Simon, Jerome Friedman, and Trevor Hastie. A blockwise descent algorithm for group-penalized multiresponse and multinomial regression. *arXiv preprint arXiv:1311.6529*, 2013. 3

[42] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5

[43] Edwin Simpson and Iryna Gurevych. Scalable bayesian preference learning for crowds. *Machine Learning*, 2020. 2

[44] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *ICML*, 2019. 1, 5, 7

[45] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017. 5

[46] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *CVPR*, 2018. 2, 3

[47] Ryutaro Tanno, Ardavan Saeedi, Swami Sankaranarayanan, Daniel C Alexander, and Nathan Silberman. Learning from noisy labels by regularized estimation of annotator confusion. In *CVPR*, 2019. 2

[48] Sunil Thulasidasan, Tanmoy Bhattacharya, Jeff Bilmes, Gopinath Chennupati, and Jamal Mohd-Yusof. Combating label noise in deep learning using abstention. In *ICML*, 2019. 2

[49] Arash Vahdat. Toward robustness against label noise in training deep discriminative neural networks. *NeurIPS*, 2017. 3

[50] Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. Learning from noisy large-scale datasets with minimal supervision. In *CVPR*, 2017. 3

[51] Martin J Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1 -constrained quadratic programming (lasso). *IEEE transactions on information theory*, 2009. 2, 5

[52] Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia. Iterative learning with open-set noisy labels. In *CVPR*, 2018. 2, 6

[53] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *ICCV*, 2019. 1, 2, 6

[54] Yikai Wang, Chengming Xu, Chen Liu, Li Zhang, and Yanwei Fu. Instance credibility inference for few-shot learning. In *CVPR*, 2020. 2

[55] Yikai Wang, Li Zhang, Yuan Yao, and Yanwei Fu. How to trust unlabeled data? instance credibility inference for few-shot learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[56] John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009. 2

[57] Pengxiang Wu, Songzhu Zheng, Mayank Goswami, Dimitris N Metaxas, and Chao Chen. A topological filter for learning with label noise. *NeurIPS*, 2020. 1, 2, 6

[58] Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. Robust early-learning: Hindering the memorization of noisy labels. In *ICLR*, 2021. 2

[59] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *CVPR*, 2015. 1, 2, 3

[60] Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. In *CVPR*, 2019. 3, 6

[61] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *ICML*, 2019. 1, 2, 6

[62] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 2, 4

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

[64] Yikai Zhang, Songzhu Zheng, Pengxiang Wu, Mayank Goswami, and Chao Chen. Learning with feature-dependent label noise: A progressive approach. In *ICLR*, 2021. 7

[65] Zhilu Zhang and Mert R Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*, 2018. 1, 2, 6

[66] Peng Zhao and Bin Yu. On model selection consistency of lasso. *Journal of Machine learning research*, 2006. 2, 5

[67] Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. Robust curriculum learning: From clean label detection to noisy label self-correction. In *ICLR*, 2021. 1, 2

[68] Xiong Zhou, Xianming Liu, Junjun Jiang, Xin Gao, and Xiangyang Ji. Asymmetric loss functions for learning with noisy labels. In *ICML*, 2021. 2

[69] Xiong Zhou, Xianming Liu, Chenyang Wang, Deming Zhai, Junjun Jiang, and Xiangyang Ji. Learning with noisy labels via sparse regularization. In *ICCV*, 2021. 2, 4, 6, 7