Self-Paced Robust Learning for Leveraging Clean Labels in Noisy Data

Xuchao Zhang,[1] Xian Wu,[2] Fanglan Chen,[1] Liang Zhao,[3] Chang-Tien Lu[1]
[1]Discovery Analytics Center, Virginia Tech, Falls Church, VA,
[2]University of Notre Dame, Notre Dame, IN,
[3]George Mason University, Fairfax, VA
[1]{xuczhang, fanglanc, ctlu}@vt.edu, [2]xwu9@nd.edu, [3]lzhao9@gmu.edu

Abstract

The success of training accurate models strongly depends on the availability of a sufficient collection of precisely labeled data. However, real-world datasets contain erroneously labeled data samples that substantially hinder the performance of machine learning models. Meanwhile, well-labeled data is usually expensive to obtain and only a limited amount is available for training. In this paper, we consider the problem of training a robust model by using large-scale noisy data in conjunction with a small set of clean data. To leverage the information contained via the clean labels, we propose a novel self-paced robust learning algorithm (*SPRL*) that trains the model in a process from more reliable (clean) data instances to less reliable (noisy) ones under the supervision of well-labeled data. The self-paced learning process hedges the risk of selecting corrupted data into the training set. Moreover, theoretical analyses on the convergence of the proposed algorithm are provided under mild assumptions. Extensive experiments on synthetic and real-world datasets demonstrate that our proposed approach can achieve a considerable improvement in effectiveness and robustness to existing methods.

1 Introduction

The availability of well-labeled data is becoming a key factor for training a machine learning model with high complexity, such as deep neural networks (LeCun, Bengio, and Hinton 2015). However, collecting a large-scale dataset with clean labels usually requires cumbersome verification work by human beings, which is time-consuming and expensive. Usually it is much easier to obtain a small set of data precisely labeled by human experts and collect large quantities of noisy labels by using crowd-sourcing (Brabham 2008) or "weak annotators" based on heuristics (Branson, Perona, and Belongie 2011). For example, to collect the training images of certain keywords, one possible solution using a "weak annotator" is to search with the keyword in an image search engine such as *Google Images*[1]. However, the images collected by searching for a polysemous word such as "apple" will show not only the fruit, but also the logo or products of *Apple* company. Although it is simple and inexpensive for a so-called "weak annotator" to collect a massive amount of training data, the training samples will inevitably contain a large amount of noise.

Leveraging the prior knowledge of clean labels in noisy data is actually a crucial issue in practice, but existing robust learning methods (McWilliams et al. 2014; Zhang et al. 2017b) typically focus more on eliminating noisy data. Also, the data collected by "weak annotator" or crowd-sourcing can be too noisy for existing robust methods to train an accurate model. Moreover, existing works that utilize additional clean labels are usually designed for specific tasks such as image classification (Zhang et al. 2017a; Veit et al. 2017). These methods typically utilize clean labels in large-scale noisy data based on their additional domain knowledge (Jiang et al. 2017; Li et al. 2017b); however, these approaches are difficult to handle extremely noisy data and heavily relied on their domain knowledge (Li et al. 2017b), which makes them difficult be used in more general problems. For those seeking to address these issues, the major challenges can be summarized as follows. 1) **Infeasibility to train an accurate model by using clean or noisy data individually.** Training models in a small amount of well-labeled data will usually cause the overfitting problem and result in an inaccurate model. On the other hand, weakly labeled data may contain a large amount of corrupted data, e.g., more than 50% data can be corrupted, which makes it hard for most of the robust learning approaches to resist the noise. 2) **Difficulty to learn models from noisy data under the supervision of a small amount of clean labels.** Models trained in a small amount of clean data is highly prone to overfitting and bias. If the biased model is directly applied to identify the uncorrupted samples in noisy data, some corrupted samples can be mistakenly included in the training set. 3) **Lack of domain knowledge to utilize clean labels to extract the uncorrupted samples from noisy data.** Most of the existing work using additional clean labels depends on extra domain knowledge, such as utilizing the information of label relations in a knowledge graph (Li et al. 2017b) for image classification task. However, such domain knowledge is usually hard to obtain in practice and prevents these methods to be more extensively used in general situations.

[1]https://images.google.com/

To address all these technical challenges, this paper presents a novel self-paced robust learning approach, named *SPRL*, to leverage the clean labels in noisy data by a self-paced training process based on samples in a nearly optimal sequence on the noisiness. The main contributions of this paper are as follows: 1) **Formulating a framework to leverage the clean labels in noisy data.** A framework is proposed to utilize clean labels in a large-scale noisy dataset. Specifically, the clean labels are assumed to contain a limited size of data while the noisy dataset may contain an extremely large amount of data corruption. The presented approach to tackle robust regression and classification tasks can be generally used in various tasks. 2) **Proposing a self-paced robust learning algorithm to train models under the supervision of clean labels.** The proposed algorithm learns the data samples from clean to noisy under the supervision of clean labels, which hedges the risk of involving corrupted data into the training set. Furthermore, the algorithm learns the training data in an order that are dynamically determined by the feedback of the learner itself without additional prior knowledge, which makes it extensively utilized in practice. 3) **Providing a theoretical analysis on the convergence of the proposed algorithm.** We prove that our self-paced robust learning algorithm converges under the assumption that the loss function selected for the estimated model has a finite lower bound. Specifically, the objective function of our algorithm monotonically decreases in accord with the increasing learning pace parameter until it reaches the lower bound. 4) **Conducting extensive experiments for performance evaluations.** The proposed method was evaluated on both synthetic data and real-world datasets in robust regression and classification tasks with different corruption and data-size settings. The results demonstrate that our approach consistently outperforms existing methods along multiple metrics. To the best of our knowledge, this is the first work to train models from noisy dataset by utilizing clean labels in a self-paced learning process.

The rest of this paper is organized as follows. Section 2 introduces the problem formulation. The proposed self-paced robust learning algorithm is presented in Section 3. Experiments on synthetic and real-world datasets are presented in Section 4. Section 5 reviews related work and the paper concludes with a summary of the research in Section 6.

2 Problem Formulation

In the setting of self-paced robust learning for leveraging clean labels in noisy data, we consider the samples to be provided in two parts with different qualities: a small set of well-labeled samples with little data corruption $\mathcal{D}_s = \big\{(\boldsymbol{x}_1, y_1)\ ,\ldots,(\boldsymbol{x}_k, y_k)\big\}$ and a weakly labeled dataset $\mathcal{D}_w = \{(\boldsymbol{x}_{k+1}, y_{k+1}),\ldots,(\boldsymbol{x}_n, y_n)\}$ in large size, where $\boldsymbol{x}_i \in \mathcal{R}^p$ represents the i^{th} sample data and y_i is the corresponding label. We assume the well-labeled data contains k samples and the weakly labeled data has $n-k$ samples, where n is much larger than k ($n \gg k$).

The goal of our study is to infer the model parameter $\boldsymbol{w} \in \mathcal{R}^p$ based on the uncorrupted data samples $\mathcal{D}^+ = \mathcal{D}_s \cup \mathcal{D}_w^+$, where the set $\mathcal{D}_w^+$ represents the uncorrupted data samples in the weakly labeled dataset $\mathcal{D}_w$, in which the samples are correctly labeled. All the samples in $\mathcal{D}_s$ are uncorrupted and correctly labeled. Therefore, our purpose is to fully utilize the uncorrupted data samples in both the clean set $\mathcal{D}_s$ and the noisy set $\mathcal{D}_w$, which can be formalized as follows:

$$\hat{\boldsymbol{w}} = \mathop{\arg\min}_{\boldsymbol{w}\in\mathcal{R}^p} \sum_{i\in\mathcal{D}_s\cup\mathcal{D}_w^+} \mathcal{L}\big(y_i, f(\boldsymbol{x}_i,\boldsymbol{w})\big) + \psi(\boldsymbol{w}), \tag{1}$$

where $\mathcal{L}$ is the loss function to measure the error between label y_i and the estimated value from model f. For example, the linear model $f(\boldsymbol{x}_i,\boldsymbol{w})\big) = \boldsymbol{x}_i^T\boldsymbol{w}$ and squared loss $\|y_i - f(\boldsymbol{x}_i,\boldsymbol{w})\big)\|_2^2$ can be applied to a regression problem. The regularization term $\psi(\boldsymbol{w})$ restricts the complexity of model parameters $\boldsymbol{w}$. The notations used in this paper are summarized in Table 1.

Table 1: Math Notations

Notations	Explanations
p	number of feature in data matrix X
k	number of samples in the clean dataset
n	number of samples in the entire dataset
$X, \boldsymbol{y}$	data matrix and its corresponding label vector
$\boldsymbol{w}$	parameters of estimated model
$\tilde{\boldsymbol{w}}$	model parameter trained in the clean set
$\boldsymbol{v}$	instance weight vector
λ	parameter to control the learning pace
μ	step size of parameter λ
$\mathcal{L}$	loss function of estimated model

The problem defined in Equation (1) is challenging in the following two aspects. First, the uncorrupted set $\mathcal{D}_w^+$ in the weakly labeled set $\mathcal{D}_w$ is unknown. Simply ignoring the whole noisy data $\mathcal{D}_w$ is not a proper solution because the amount of well-labeled data is not large enough to train an accurate model. Second, the weakly labeled data $\mathcal{D}_w$ can be extremely noisy or even contain adversarial data corruption, which makes the uncorrupted samples in $\mathcal{D}_w$ difficult to estimate. In the next section, we present a self-paced-learning approach to address these challenges.

3 Proposed Method

In Section 3.1, we propose the *SPRL* algorithm to utilize clean labels in training noisy datasets. Then, the convergence analysis of our algorithm is presented in Section 3.2.

3.1 SPRL Algorithm

In order to utilize the prior knowledge on the high-quality data $\mathcal{D}_s$ on the weakly labeled data $\mathcal{D}_w$, we reformulate our objective function $\mathcal{J}$ as follows:

$$\mathop{\arg\min}_{\boldsymbol{w}\in\mathcal{R}^n,\boldsymbol{v}\in[0,1]} \mathcal{J}(\boldsymbol{w},\boldsymbol{v};\lambda) = \sum_{i=1}^{k}\mathcal{L}\big(y_i, f(\boldsymbol{x}_i,\boldsymbol{w})\big) + \sum_{i=k+1}^{n} v_i\mathcal{L}\big(y_i, f(\boldsymbol{x}_i,\boldsymbol{w})\big) + \|\boldsymbol{w}\|_2^2 + \theta\|\boldsymbol{w}-\tilde{\boldsymbol{w}}\|_2^2 - \lambda\sum_{i=k+1}^{n} v_i, \tag{2}$$

Algorithm 1: SPRL ALGORITHM

```
   Input: $X \in \mathcal{R}^{p\times n}$, $\boldsymbol{y} \in \mathcal{R}^n$, $\theta \in \mathcal{R}$, $\lambda^0 \in \mathcal{R}$, $\lambda_\infty \in \mathcal{R}$,
          $\mu \in \mathcal{R}$
   Output: solution $\boldsymbol{w}^{(t+1)}$, $\boldsymbol{v}^{(t+1)}$
1  $\tilde{\boldsymbol{w}} \leftarrow \arg\min_{\boldsymbol{w}} \sum_{i=1}^{k} \mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w})\big) + \psi(\boldsymbol{w})$
2  Initialize $\boldsymbol{w}^0 = \tilde{\boldsymbol{w}}$, $\varepsilon > 0$, $t \leftarrow 0$
3  repeat
4      for $i = k+1 \ldots n$ do
5          $v_i^{t+1} \leftarrow \infty\Big(\mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w}^t)\big) < \lambda^t\Big)$
6      Update $\boldsymbol{w}^{t+1}$ by Equation (6) with fixed $\boldsymbol{v}^{t+1}$ and
       $\tilde{\boldsymbol{w}}$.
7      $\lambda^{t+1} \leftarrow \lambda^t * \mu$
8      if $\lambda^{t+1} > \lambda_\infty$ then
9          $\lambda^{t+1} \leftarrow \lambda_\infty$
10     $t \leftarrow t+1$
11 until $\|\mathcal{J}(\boldsymbol{w}^{t+1}, \boldsymbol{v}^{t+1}; \lambda^{t+1}) - \mathcal{J}(\boldsymbol{w}^t, \boldsymbol{v}^t; \lambda^t)\|_2 < \varepsilon$
12 return $\boldsymbol{w}^{t+1}$, $\boldsymbol{v}^{t+1}$
```

where $\mathcal{L}$ represents the loss function and variable v_i is denoted as the weight of the ith data instance, which is allowed to take any value in the interval $[0, 1]$. The first term is the total loss of the clean set $\mathcal{D}_s$, in which we assume all the samples in clean set are included into our training set. The second term represents the total loss of noisy set $\mathcal{D}_w$, where the samples are selected by the instance weight vector $\boldsymbol{v}$. The term $\|\boldsymbol{w}\|_2^2$ is to control the complexity of model parameters $\boldsymbol{w}$. The variable $\tilde{\boldsymbol{w}}$ represents the model weights trained by the clean set $\mathcal{D}_s$. Since the clean set is assumed to contain few data corruption, we can minimize the following objective function without considering its data noises:

$$\tilde{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{k} \mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w})\big) + \psi(\boldsymbol{w}), \tag{3}$$

where $\psi(\boldsymbol{w})$ is the regularization term to control the complexity of $\tilde{\boldsymbol{w}}$. The term $\theta\|\boldsymbol{w} - \tilde{\boldsymbol{w}}\|_2^2$ is to control the difference between the estimated model and the model $\hat{\boldsymbol{w}}$ trained in the clean set only. If a larger parameter θ is given, the algorithm prefers to give a similar estimated model as the $\hat{\boldsymbol{w}}$. The last term $\lambda \sum_{i=k+1}^{n} v_i$ is the regularization term for instance weights $\boldsymbol{v}$, where parameter λ controls the learning pace.

The problem defined in Equation (2) can be solved based on alternate convex search (*ACS*) method (Gorski, Pfeuffer, and Klamroth 2007). When model parameter $\boldsymbol{w}$ is fixed, the variable $\boldsymbol{v}$ in the iteration t+1 can be solved by the following sub-problem:

$$v_i^{t+1} = \arg\min_{v_i \in [0,1]} \sum_{i=k+1}^{n} v_i \mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w}^t)\big) - \lambda^t \sum_{i=k+1}^{n} v_i, \tag{4}$$

where λ^t represents the value of λ in the tth iteration. A closed-form solution of $\boldsymbol{v}^{t+1}$ can be given as follows:

$$v_i^{t+1} = \begin{cases} 1, & \text{if } \mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w}^t)\big) < \lambda^t \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

When we fix the value of variable $\boldsymbol{v}^{t+1}$, variable $\boldsymbol{w}^{t+1}$ can be solved by the following sub-problem:

$$\begin{aligned} \boldsymbol{w}^{t+1} = \arg\min_{\boldsymbol{w} \in \mathcal{R}^p} & \sum_{i=1}^{k} \mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w})\big) \\ + \sum_{i=k+1}^{n} v_i^{t+1} \mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w})\big) & + \|\boldsymbol{w}\|_2^2 + \theta\|\boldsymbol{w} - \tilde{\boldsymbol{w}}\|_2^2 \end{aligned} \tag{6}$$

The sub-problem can be easily solved by an off-the-shelf optimizer when the loss function is convex. In traditional self-paced learning, the value of variable λ is increased until the objective function is converged. However, in robust learning, when λ grows, more corrupted samples with larger losses will be gradually added into the training set. Even a few corrupted samples can lead to an extremely large loss if the value of λ is too large. Therefore, in order to keep the corrupted data out of our training set, we introduce a threshold parameter λ_∞ to control the size of training set.

The details of the *SPRL* algorithm are presented in Algorithm 1. In Line 1, the model parameter $\tilde{\boldsymbol{w}}$ is trained in the clean set. In Lines 2, the variable $\boldsymbol{w}^0$ is initialized as the value of $\tilde{\boldsymbol{w}}$. The variable $\boldsymbol{v}^{t+1}$ is updated in Line 5 with variable $\boldsymbol{w}^t$ fixed. After which, the model parameter $\boldsymbol{w}^{t+1}$ is optimized by Equation (6) in Line 6 with the variables $\boldsymbol{v}^{t+1}$ fixed. In Lines 8-11, the parameter λ is enlarged to include more data instances in the training set, where λ_∞ is the threshold parameter for λ and μ is the step size. The algorithm will be stopped when the objective function is converged in Line 13. The convergence of the algorithm will be proved in Section 3.2.

3.2 Convergence Analysis

In this section, we will present the theoretical analysis on the convergence of the proposed algorithm.

First, the assumption on the loss function $\mathcal{L}$ is as follows.

Assumption 1 (Lower Bound)**.** *The loss function $\mathcal{L}$ in problem* (2) *has a lower bound $\mathcal{B}$ as follows:*

$$\mathcal{B} = \min_{\boldsymbol{w}} \mathcal{L}\big(y, f(\boldsymbol{x}, \boldsymbol{w})\big) > -\infty \tag{7}$$

This assumption can be easily satisfied by most loss functions; e.g., least-squares loss and hinge loss have their lower bound $\mathcal{B} = 0$. Then we can get the following property of the objective function:

Lemma 1. *The objective function $\mathcal{J}$ in Equation* (2) *is lower bounded as follows:*

$$\lim_{t\to\infty} \mathcal{J}(\boldsymbol{w}^t, \boldsymbol{v}^t; \lambda^t) > -\infty \tag{8}$$

Proof. The objective function $\mathcal{J}$ has the following property:

$$\begin{aligned}
&\mathcal{J}(\boldsymbol{w}^t, \boldsymbol{v}^t; \lambda^t) \\
&\quad \overset{(a)}{\geq} \sum_{i=1}^{k} \mathcal{B} + \sum_{i=k+1}^{n} v_i^t \mathcal{B} + \|\boldsymbol{w}^t\|_2^2 + \theta \|\boldsymbol{w}^t - \tilde{\boldsymbol{w}}\|_2^2 \\
&\quad - \lambda^t \sum_{i=k+1}^{n} v_i^t \overset{(b)}{\geq} k\mathcal{B} + \sum_{i=k+1}^{n} v_i^t \mathcal{B} - (n-k) \cdot \lambda_\infty
\end{aligned} \tag{9}$$

Inequality (a) follows from $\mathcal{L}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{w}^t) \geq \mathcal{B}$ and $v_i^t \geq 0$. Inequality (b) follows from $\|\boldsymbol{w}^t\|_2^2 \geq 0$, $\theta\|\boldsymbol{w}^t - \tilde{\boldsymbol{w}}\|_2^2 \geq 0$, and λ_∞ is the maximum threshold of parameter λ. When all the v_i^t are equal to 1, $\lambda^t \sum_{i=k+1}^{n} v_i^t$ reaches its minimum value $(n-k) \cdot \lambda_\infty$. When lower bound $B \geq 0$, we have $\sum_{i=k+1}^{n} v_i^t \mathcal{B} \geq 0$; otherwise, when $B < 0$, we have $\sum_{i=k+1}^{n} v_i^t \mathcal{B} \geq (n-k) \cdot \mathcal{B}$. Therefore, we have

$$\begin{aligned}
\mathcal{J}(\boldsymbol{w}^t, \boldsymbol{v}^t; \lambda^t) &\geq k\mathcal{B} + \min\big\{0, (n-k) \cdot \mathcal{B}\big\} - (n-k) \cdot \lambda_\infty \\
&= k\mathcal{B} + (n-k) \cdot \big(\min\big\{0, \cdot \mathcal{B}\big\} - \lambda_\infty\big)
\end{aligned} \tag{10}$$

Since $\mathcal{B} > -\infty$ and λ_∞ is constant, we have $\mathcal{J}(\boldsymbol{w}^t, \boldsymbol{v}^t; \lambda^t) > -\infty$ for $\forall t = 1 \ldots \infty$. □

Theorem 1. *When Assumption 1 is satisfied, Algorithm 1 converges with the following property:*

$$\lim_{t \to \infty} \|\mathcal{J}^{t+1} - \mathcal{J}^t\|_2 = 0 \tag{11}$$

The proof of Theorem 1 can be found in supplementary document: https://xuczhang.github.io/papers/aaai20_sprl_supp.pdf

4 Experiment

In this section, the proposed *SPRL* algorithm is evaluated on synthetic and real-world datasets in robust regression and classification tasks. After the experiment setup has been introduced in Section 4.1, we present results on the robust regression task compared against several existing methods on both synthetic and real-world datasets in Section 4.2, along with performance results for the binary classification task in Section 4.3. The analysis on parameter λ is presented in supplementary document. All the experiments were conducted on a 64-bit machine with an Intel(R) Core(TM) quad-core processor (i7CPU@3.6GHz) and 32.0GB of memory. Details of both the source code and datasets used in the experiments can be found in supplementary document.

4.1 Experiment Setup

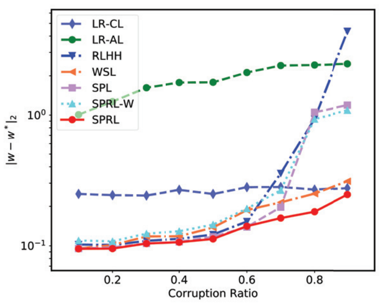
Datasets and Labels Our datasets consist of synthetic and real-world data. The simulation samples for linear regression were randomly generated according to the model $\boldsymbol{y} = X^T \boldsymbol{w}_* + \boldsymbol{u} + \boldsymbol{\varepsilon}$, where $\boldsymbol{w}_*$ is the ground truth coefficients and $\boldsymbol{u}$ is the adversarial corruption vector. $\boldsymbol{\varepsilon}$ represents the additive dense noise, where $\varepsilon_j \sim \mathcal{N}(0, \sigma^2)$. We sampled the ground-truth regression coefficients $\boldsymbol{w}_* \in \mathcal{R}^p$ as a random unit norm vector. The covariance data X was drawn independently and identically distributed from $\boldsymbol{x}_i \sim \mathcal{N}(0, I_p)$ where I_p is identity matrix, and the uncorrupted response variables were generated as $\boldsymbol{y}_* = X^T \boldsymbol{w}_* + \boldsymbol{\varepsilon}$. The corrupted response vector was generated as $\boldsymbol{y} = \boldsymbol{y}_* + \boldsymbol{u}$, where the corruption vector $\boldsymbol{u}$ was sampled from the uniform distribution $[-5\|\boldsymbol{y}_*\|_\infty, 5\|\boldsymbol{y}_*\|_\infty]$. The set of uncorrupted points was selected as a uniformly random subset of $[n]$. Similarly, the authentic samples for the binary classification task were generated according to the model $\boldsymbol{y}_* = \operatorname{sign}(X^T \boldsymbol{w}_* + \boldsymbol{\varepsilon})$, where $\boldsymbol{\varepsilon}$ is the additive dense noise. The labels of corrupted samples were set as the opposite values in $\boldsymbol{y}_*$.

For the real-world datasets, we chose the *BlogFeedback* dataset (Buza 2014) for the robust regression task to predict the number of blog comments in the upcoming 24 hours. The data originates from blog posts with 280 feature attributes, including the total number of comments before base time and the number of comments in the first 24 hours after the publication of the blog post. We chose the first 21,000 data samples as the training set, in which the clean set contains 1,000 samples and the noisy set contains the remaining 20,000 samples. Additional data corruption was sampled from the uniform distribution $[-0.5y_i, 0.5y_i]$, where y_i denotes the blog feedback number of the i^{th} sample data. For the classification task, we chose the Large Movie Review dataset (Maas et al. 2011) collected from the IMDb website[2] for sentiment classification of movie reviews. The first 12,000 data samples were used as training set, of which 2,000 samples were used as the clean set and the remaining samples as the noisy set. Additional data corruption was also added into the noisy set by reversing their labels. The other 10,000 data samples were chosen as the testing set. The features were represented as the tf-idf value of each word. One thousand features with the largest tf-idf values were chosen in our experiment.

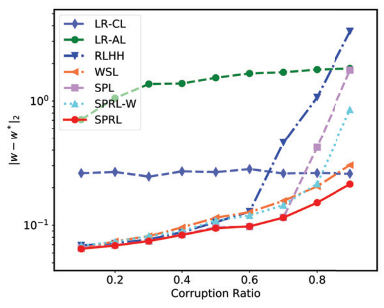
Evaluation Metrics For the robust regression task, we measured the performance of the regression coefficients recovery using the L_2 error $e = \|\hat{\boldsymbol{w}} - \boldsymbol{w}_*\|_2$ in the synthetic data, where $\hat{\boldsymbol{w}}$ represents the estimated coefficients for each compared method and $\boldsymbol{w}_*$ is the ground truth regression coefficients. For the BlogFeedback dataset, we use the root-mean-squared-error (*RMSE*) to evaluate the performance of blog feedback prediction. Defining $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}_*$ as the predicted feedback number and ground truth number, respectively, the root-mean-squared-error can be presented as $\mathrm{RMSE}(\hat{\boldsymbol{y}}, \boldsymbol{y}_*) = \frac{1}{n}\|\hat{\boldsymbol{y}} - \boldsymbol{y}_*\|_2$, where n is the number of samples. To validate the performance for binary classification, precision, recall, and F1-score are measured by comparing the estimated labels with the actual ones on both the synthetic and real-world datasets.

Comparison Methods We use the following methods to evaluate the performance of our algorithm in the robust regression tasks: 1) The *Linear Regression on Clean Data* (*LR-CL*) method takes the linear regression on the clean data only without using the noisy data. 2) The *Linear Regression on All Data* (*LR-AL*) method takes the linear regression on all the data without using the prior knowledge of the clean labels. 3) The method *RLHH* (Zhang et al. 2017b) applies

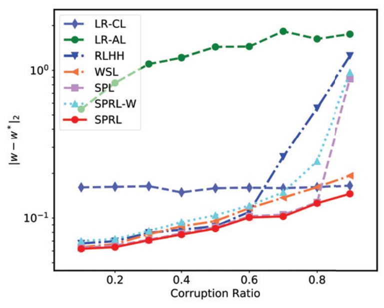
[2]http://www.imdb.com



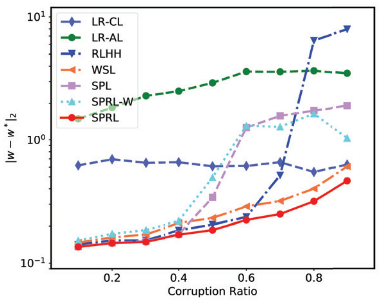
(a) p=400, k=1K, n=6K, dense noise



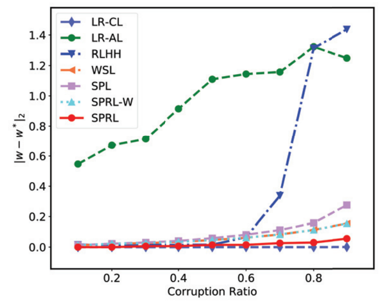
(b) p=400, k=1K, n=11K, dense noise



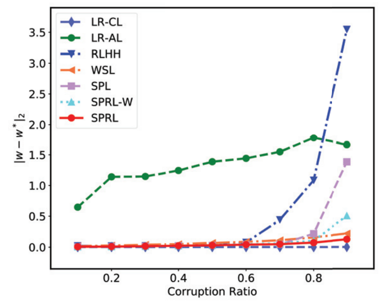
(c) p=400, k=2K, n=12K, dense noise



(d) p=800, k=1K, n=6K, dense noise



(e) p=200, k=1K, n=11K, no dense noise



(f) p=400, k=1K, n=11K, no dense noise

Figure 1: Performance on Regression Coefficient Recovery for Different Corruption Ratios in Uniform Distribution.

Table 2: Mean Absolute Error of Blog Feedback Prediction

	Corruption Ratio					
	10%	**30%**	**50%**	**70%**	**90%**	**Avg.**
LR-CL	1.159	1.161	1.153	1.164	1.173	1.162
LR-AL	7.254	17.116	10.459	17.226	8.334	12.0778
WSL	0.981	1.280	2.562	2.154	1.375	1.6704
SPL	0.973	1.189	3.666	4.382	4.525	2.947
SPRL-W	**0.919**	2.627	2.493	4.547	5.797	3.2766
SPRL	0.971	**1.107**	**1.036**	**1.053**	**1.046**	**1.0426**

a recently proposed heuristic hard-thresholding based robust method, in which the entire dataset was directly applied without utilizing the prior knowledge of clean labels. 4) A variant of the weakly supervised learning method (Jiang et al. 2017) (*WSL*) is applied to estimate the uncorrupted samples of the noisy set under the supervision of clean labels by the feedback of the loss function. The method can also be regarded as a non-paced version of *SPRL*. 5) The traditional self-paced learning algorithm (*SPL*) (Kumar, Packer, and Koller 2010) was also compared in our experiments with the parameter $\lambda = 0.1$ and the step size $\mu = 1.1$. 6) *SPRL-W* is a variant of our proposed method *SPRL* without using the clean data set in its objective function. In particular, the first term in Equation (2) is omitted. To evaluate the performance in the binary classification task, we applied the following competing methods: 1) The *SVM on Clean Data* (*SVM-CL*) method uses the binary support vector machine on the clean data only without using the noisy data. 2) The *SVM on All Data* (*SVM-AL*) method takes the support vector machine method on the entire dataset without using the prior knowledge of the clean dataset. 3) A recently proposed robust logistic regression algorithm, called *RoLR* (Feng et al. 2014), which estimates the parameter through a linear programming procedure, was also compared. 4) A similar *WSL* method using the *SVM* loss function was also evaluated in the robust classification task. 5) The self-paced learning algorithm (*SPL*) (Kumar, Packer, and Koller 2010) was also compared in the robust classification task with the initial parameter $\lambda = 0.1$ and the step size $\mu = 1.1$. 6) *SPRL-W* using the classification loss function was also compared with the same setting as the *SPL* method. For our proposed method, *SPRL*, we choose the parameter λ_∞ as 1 and 3.5 for regression and classification tasks, respectively. All the results from comparison methods were averaged over 10 runs.

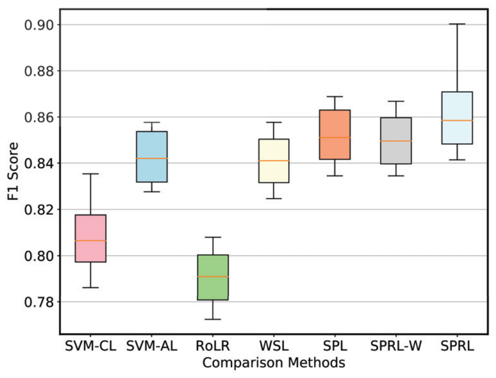
4.2 Performance on Robust Regression

Regression Coefficient Recovery on Synthetic Data Figure 1 shows coefficient recovery performance for different settings on corruption ratios and data sizes. Specifically, Figure 1a and Figure 1b present the recovery performance for different amounts of noisy data with the same amounts of features and clean data. Looking at the results, we can conclude: 1) The *SPRL* methed outperforms all the competing methods, including the traditional self-paced learning method with the same parameter settings. 2) *LR-CL* has stable performance because the clean data is not affected by the change of corruption ratio. However, if a model is merely

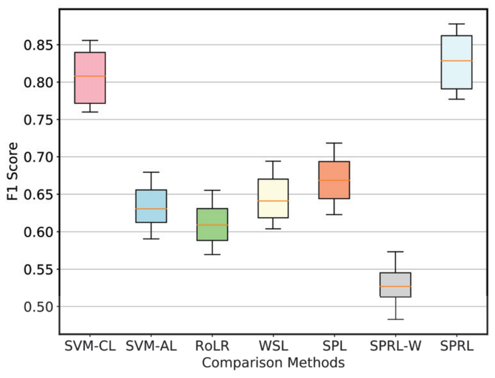
Table 3: Performance on Binary Classification (F1, Precision, Recall)

	feature=200, clean set=100, noisy set=5K				**feature=400, clean set=100, noisy set=5K**			
	10%	20%	30%	40%	10%	20%	30%	40%
SVM-CL	0.657,0.656,0.659	0.654,0.650,0.658	0.676,0.667,0.686	0.674,0.688,0.661	0.628,0.629,0.628	0.639,0.640,0.638	0.626,0.621,0.630	0.620,0.627,0.613
SVM-AL	0.928,0.927,0.929	0.900,0.902,0.898	0.835,0.831,0.838	0.750,**0.754**,0.747	0.918,0.916,0.920	0.860,0.861,0.859	**0.786,0.796**,0.776	0.665,0.670,0.661
RoLR	0.814,0.817,0.810	0.842,0.840,0.845	0.804,0.795,0.814	0.724,0.730,0.719	0.827,0.834,0.820	0.788,0.790,0.785	0.747,0.758,0.736	0.650,0.659,0.641
WSL	0.886,0.889,0.883	0.791,0.792,0.789	0.745,0.739,0.752	0.706,0.715,0.697	0.870,0.873,0.868	0.786,0.789,0.783	0.690,0.690,0.690	0.644,0.653,0.635
SPL	0.946,0.946,0.946	0.903,0.905,0.902	0.809,0.805,0.813	0.665,0.666,0.665	0.916,0.921,0.912	0.824,0.822,0.826	0.739,0.744,0.735	0.608,0.614,0.602
SPRL-W	0.944,0.942,0.946	0.913,0.916,0.910	0.799,0.796,0.802	0.694,0.699,0.689	0.905,0.903,0.906	0.811,0.815,0.808	0.754,0.760,0.749	0.637,0.647,0.628
SPRL	**0.968**,**0.965**,**0.971**	**0.922**,**0.918**,**0.928**	**0.871**,**0.874**,**0.866**	**0.751**,0.742,**0.754**	**0.935**,**0.936**,**0.932**	**0.864**,**0.863**,**0.865**	0.780,0.785,**0.783**	**0.681**,**0.691**,**0.674**

	feature=200, clean set=200, noisy set=5K				**feature=200, clean set=200, noisy set=10K**			
	10%	20%	30%	40%	10%	20%	30%	40%
SVM-CL	0.758,0.756,0.759	0.722,0.720,0.725	0.734,0.730,0.739	0.734,0.738,0.730	0.715,0.718,0.712	0.730,0.734,0.726	0.732,0.728,0.736	0.701,0.697,0.705
SVM-AL	0.942,0.939,0.944	0.897,0.891,0.904	0.853,0.846,0.861	0.749,0.743,0.756	0.948,0.946,0.950	0.932,0.934,0.930	0.898,0.899,0.897	0.787,0.790,0.784
RoLR	0.833,0.833,0.834	0.834,0.834,0.834	0.808,0.806,0.811	0.699,0.693,0.705	0.879,0.877,0.882	0.886,0.884,0.888	0.665,0.668,0.662	0.771,0.770,0.771
WSL	0.905,0.899,0.911	0.827,0.825,0.829	0.796,0.794,0.798	0.743,0.747,0.740	0.902,0.900,0.904	0.856,0.861,0.851	0.798,0.801,0.795	0.722,0.718,0.727
SPL	0.950,0.951,0.949	0.905,0.899,0.912	0.810,0.810,0.810	0.665,0.665,0.665	0.967,0.965,0.969	0.959,0.963,0.954	0.869,0.875,0.864	0.687,0.689,0.686
SPRL-W	0.949,0.949,0.949	0.896,0.892,0.900	0.822,0.823,0.821	0.745,0.736,0.755	0.966,0.964,0.969	0.950,0.953,0.946	0.902,0.903,0.900	0.721,0.722,0.721
SPRL	**0.963**,**0.967**,**0.960**	**0.926**,**0.925**,**0.927**	**0.876**,**0.878**,**0.875**	**0.768**,**0.780**,**0.763**	**0.981**,**0.977**,**0.985**	**0.959**,**0.954**,**0.963**	**0.920**,**0.928**,**0.912**	**0.787**,**0.782**,**0.793**



(a) Clean set=2K, Noisy set=10K, Corruption Ratio=10%



(b) Clean set=2K, Noisy set=10K, Corruption Ratio=50%

Figure 2: Sentiment Classification Performance on Movie Reviews

trained in clean data with a small data size, the performance is almost two times worse than that of *SPRL*. When the corruption ratio is larger than 80%, it outperforms the other competing methods except for *SPRL* because little uncorrupted data is contained in the noisy set. 3) *LR-AL* always performs worse than the other methods since it does not consider the data corruption and the prior knowledge on the clean set. 4) *RLHH* has a competitive performance when the data corruption is less than 50%, however, when the corruption ratio is over 60%, the recovery error increases dramatically. 5) The recovery errors of *SPL* and *SPRL-W* also dramatically increased when the corruption ratio was over 60% because these two methods do not properly utilize the prior knowledge of the clean set. 6) The *WSL* method performs around 10% worse than *SPRL*, which shows the self-paced training process can improve performance in the robust learning problem. When the size of the clean set increases, Figure 1c shows similar results as Figure 1a with a decreased overall recovery error. When the number of features increases in Figure 1d, the overall recovery error increased around 200% on average for all the methods. Figures 1e and 1f present the performance in a no-dense-noise setting. Since *LR-CL* has no impact on different corruption ratios, it can always recover the ground truth coefficient exactly. However, *SPRL* can still outperform the other methods under different corruption ratios, achieving a close recovery of the ground truth coefficient.

Blog Feedback Prediction Table 2 shows the results of blog feedback prediction in different corruption settings. Since the *RLHH* method cannot perfectly handle data corruption greater than 40%, its result is not shown in Table 2. From the results, we can conclude: 1) *SPRL* consistently outperforms the other competing methods except when the corruption ratio is 10%, in which case most of the methods have extremely close results. 2) The error of the *SPL* and *SPRL-W* methods increase dramatically when the corruption ratio is raised. However, *SPRL* has consistent performance with little impact of the corruption ratio. 3) *WSL* has competitive results, but its error is still around 40% larger than our proposed method. 4) *LR-CL* method has good performance when the data is extremely noisy. However, our proposed method still gets 11.9% improvement when combining both clean and noisy data. 5) *LR-AL* performs the worst since it is simply applied in the entire dataset without considering the data corruption.

4.3 Performance on Robust Classification

Binary Classification on Synthetic Data Table 3 shows the results on the robust binary classification task on different settings of features and data sizes. From the results, we conclude: 1) The *SPRL* method outperforms all the competing methods consistently in different settings, including *RoLR*, whose corruption ratio parameter uses the ground truth value. 2) *SVM-CL* trained on the clean set performs worse than the other methods, which shows that utilizing the large-scale noisy data is important when the size of clean set is small. 3) *SVM-AL* has a competitive performance when the size of the clean set is not large enough for training. But our proposed method, *SPRL*, can still perform better than *SVM-AL* even when 40% of the data labels are corrupted. 4) The *SPRL-W* method usually performs better than *SPL* when the corruption ratio is larger than 20%. This is because *SPRL-W* utilizes the prior knowledge on the clean set, which greatly helps when the corruption ratio increases.

Sentiment Classification of Movie Reviews The performance on sentiment classification of IMDb movie reviews is shown in Figure 2. We use box plots to show the distribution of results based on the five number summary: minimum, first quartile, median, third quartile, and maximum. The results are run on 20 independent datasets sampled from the IMDb dataset with different corruption ratios. From the results, we can conclude: 1) *SPRL* outperforms the other competing methods in different settings of corruption ratio. 2) When the corruption ratio is small, the performance of *SVM-CL* is worse than all the other methods. However, when the corruption ratio is increased, its F1 scores are better than all the other methods except *SPRL*, which has a consistent performance. 3) When the corruption ratio increases, the performance of *SPRL-W* degrades dramatically since the clean set is not included in its training set. 4) Even when the corruption ratio is 50%, *SPRL* still performs better than *SVM-CL* because it can utilize the uncorrupted data in the noisy data to improve its performance in the training process.

5 Related Work

In this section, the work related to this paper is summarized from the following two aspects.

5.1 Self-Paced Learning

In recent years, self-paced learning (Kumar, Packer, and Koller 2010) has received widespread attention for various applications in machine learning, such as image classification (Jiang et al. 2015), event detection (Jiang et al. 2014a) and object tracking (Zhang et al. 2016). Inspired by the learning process of humans and animals (Bengio et al. 2009), self-paced learning (*SPL*) (Kumar, Packer, and Koller 2010) considers to approach training data in a meaningful order, from easy to hard, to facilitate the learning process. Unlike standard curriculum learning (Bengio et al. 2009), which learns the data in a predefined curriculum design based on prior knowledge, *SPL* learns the training data in an order that is dynamically determined by feedback during the learning process itself, which means it can be more extensively utilized in practice. Furthermore, a wide assortment of *SPL*-based methods (Pi et al. 2016; Ma et al. 2017) have been developed, including self-paced curriculum learning (Jiang et al. 2015), self-paced learning with diversity (Jiang et al. 2014b), multi-view (Xu, Tao, and Xu 2015), and multi-task (Li et al. 2017a; Keerthiram Murugesan 2017) self-paced learning. In addition, several researchers have conducted theoretical analyses of self-paced learning. Meng et al. (Meng, Zhao, and Jiang 2015) provides a theoretical analysis of the robustness of *SPL*, revealing that the implicit objective function of *SPL* has a similar configuration to a non-convex regularized penalty. Such regularization restricts the contributions of noisy data samples to the objective, and thus enhances the learning robustness. Ma et al. (Ma, Liu, and Meng 2017) proved that the learning process of *SPL* always converges to critical points of its implicit objective under mild conditions. However, none of the existing self-paced learning approaches can be applied to our problem of leveraging clean labels in noisy data.

5.2 Robust Learning

A large body of literature on the robust learning problem has been established over the last few decades. Most of the studies aim to directly learn from noisy labels and focus on noise-robust algorithms. For instance, (Chen, Caramanis, and Mannor 2013) proposed a robust algorithm based on trimmed inner product. (McWilliams et al. 2014) proposed a sub-sampling algorithm for large-scale corrupted linear regression. (Bhatia, Jain, and Kar 2015) and (Zhang et al. 2017b) proposed hard-thresholding based methods with strong guarantees of coefficient recovery under a mild assumption on datasets. Another group of methods focus on removing or correcting mislabeled data. For example, some work utilized heavy-tailed distributions (Zhu, Leung, and He 2013) such as Student t-distribution and Poisson distribution to model the mislabeled data, while others detected these outliers based on Gaussian distribution (Solberg and Lahti 2005; Hodge and Austin 2004) under the assumption that outliers have a small probability of occurrence in the population. Some methods do not assume any prior knowledge on the data distribution based on kernel functions (Latecki, Lazarevic, and Pokrajac 2007; Roth 2006). These approaches utilize kernel functions to approximate the actual density distribution and declare the instances lying in the low probability area of the kernel density function as outliers. However, all these approaches typically jointly learn the clean and noisy data together, but cannot fully leverage the information contained in the clean set.

6 Conclusion

In this paper, a self-paced robust learning algorithm is proposed to leverage clean labels in noisy data. To achieve this, the self-paced learning process selects data instances from clean to noisy under the supervision of the well-labeled data, which helps to hedge the risk of learning from corrupted data samples. Moreover, theoretical analysis shows that our

SPRL algorithm can be converged when the loss function is lower bounded. Extensive experiments on both synthetic data and real-world data on robust regression and classification tasks demonstrate that the proposed algorithm outperforms the other comparable methods over a range of different data settings.

References

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, 41–48. ACM.

Bhatia, K.; Jain, P.; and Kar, P. 2015. Robust regression via hard thresholding. In *Advances in Neural Information Processing Systems*, 721–729.

Brabham, D. C. 2008. Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence* 14(1):75–90.

Branson, S.; Perona, P.; and Belongie, S. 2011. Strong supervision from weak annotation: Interactive training of deformable part models. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 1832–1839. IEEE.

Buza, K. 2014. Feedback prediction for blogs. In Spiliopoulou, M.; Schmidt-Thieme, L.; and Janning, R., eds., *Data Analysis, Machine Learning and Knowledge Discovery*, 145–152. Cham: Springer International Publishing.

Chen, Y.; Caramanis, C.; and Mannor, S. 2013. Robust sparse regression under adversarial corruption. In Dasgupta, S., and McAllester, D., eds., *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, 774–782. Atlanta, Georgia, USA: PMLR.

Feng, J.; Xu, H.; Mannor, S.; and Yan, S. 2014. Robust logistic regression and classification. In *Advances in neural information processing systems*, 253–261.

Gorski, J.; Pfeuffer, F.; and Klamroth, K. 2007. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research* 66(3):373–407.

Hodge, V., and Austin, J. 2004. A survey of outlier detection methodologies. *Artificial intelligence review* 22(2):85–126.

Jiang, L.; Meng, D.; Mitamura, T.; and Hauptmann, A. G. 2014a. Easy samples first: Self-paced reranking for zero-example multimedia search. In *Proceedings of the 22nd ACM international conference on Multimedia*, 547–556. ACM.

Jiang, L.; Meng, D.; Yu, S.-I.; Lan, Z.; Shan, S.; and Hauptmann, A. 2014b. Self-paced learning with diversity. In *Advances in Neural Information Processing Systems*, 2078–2086.

Jiang, L.; Meng, D.; Zhao, Q.; Shan, S.; and Hauptmann, A. G. 2015. Self-paced curriculum learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, 2694–2700. AAAI Press.

Jiang, L.; Zhou, Z.; Leung, T.; Li, L.-J.; and Fei-Fei, L. 2017. Mentornet: Regularizing very deep neural networks on corrupted labels. *arXiv preprint arXiv:1712.05055*.

Keerthiram Murugesan, J. C. 2017. Self-paced multitask learning with shared knowledge. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2522–2528.

Kumar, M. P.; Packer, B.; and Koller, D. 2010. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, 1189–1197.

Latecki, L. J.; Lazarevic, A.; and Pokrajac, D. 2007. Outlier detection with kernel density functions. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 61–75. Springer.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature* 521(7553):436.

Li, C.; Yan, J.; Wei, F.; Dong, W.; Liu, Q.; and Zha, H. 2017a. Self-paced multi-task learning. In *AAAI*, 2175–2181.

Li, Y.; Yang, J.; Song, Y.; Cao, L.; Luo, J.; and Li, J. 2017b. Learning from noisy labels with distillation. *2017 IEEE International Conference on Computer Vision (ICCV)* 1928–1936.

Ma, F.; Meng, D.; Xie, Q.; Li, Z.; and Dong, X. 2017. Self-paced co-training. In *International Conference on Machine Learning*, 2275–2284.

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

Maas, A. L.; Daly, R. E.; Pham, P. T.; Huang, D.; Ng, A. Y.; and Potts, C. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, 142–150. Stroudsburg, PA, USA: Association for Computational Linguistics.

McWilliams, B.; Krummenacher, G.; Lucic, M.; and Buhmann, J. M. 2014. Fast and robust least squares estimation in corrupted linear models. In *Advances in Neural Information Processing Systems*, 415–423.

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

Pi, T.; Li, X.; Zhang, Z.; Meng, D.; Wu, F.; Xiao, J.; and Zhuang, Y. 2016. Self-paced boost learning for classification. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, 1932–1938. AAAI Press.

Roth, V. 2006. Kernel fisher discriminants for outlier detection. *Neural computation* 18(4):942–960.

Solberg, H. E., and Lahti, A. 2005. Detection of outliers in reference distributions: performance of horn's algorithm. *Clinical chemistry* 51(12):2326–2332.

Veit, A.; Alldrin, N.; Chechik, G.; Krasin, I.; Gupta, A.; and Belongie, S. 2017. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 839–847.

Xu, C.; Tao, D.; and Xu, C. 2015. Multi-view self-paced learning for clustering. In *IJCAI*, 3974–3980.

Zhang, D.; Meng, D.; Zhao, L.; and Han, J. 2016. Bridging saliency detection to weakly supervised object detection based on self-paced curriculum learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, 3538–3544. AAAI Press.

Zhang, X.; Zhao, L.; Boedihardjo, A. P.; and Lu, C. 2017a. Online and distributed robust regressions under adversarial data corruption. In *2017 IEEE International Conference on Data Mining (ICDM)*, volume 00, 625–634.

Zhang, X.; Zhao, L.; Boedihardjo, A. P.; and Lu, C.-T. 2017b. Robust regression via heuristic hard thresholding. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, IJCAI'17. AAAI Press.

Zhu, H.; Leung, H.; and He, Z. 2013. A variational bayesian approach to robust sensor fusion based on student-t distribution. *Information Sciences* 221(Supplement C):201 – 214.