Correction of Noisy Labels via Mutual Consistency Check

Sahey Bhadra[a,b,*], Matthias Hein[b]

[a] *Max-Planck Institute for Informatics, Saarbrucken, Germany*
[b] *Saarland University, Saarbrucken, Germany*

Abstract

Label noise can have severe negative effects on the performance of a classifier. Such noise can either arise by adversarial manipulation of the training data or from unskilled annotators frequently encountered in crowd sourcing (e.g. Amazon mechanical turk). Based on the assumption that an expert has provided some fraction of the training data, where labels can be assumed to be true, we propose a new pre-processing method to identify and correct noisy labels via a mutual consistency check using a Parzen window classifier. While the resulting optimization problem turns out to be a combinatorial problem, we design an efficient algorithm for which we provide approximation guarantees. Extensive experimental evaluation shows that our method performs similar and often much better than existing methods for the detection of noisy labels, thus leading to a boost in performance of the resulting classifiers.

Keywords: Noisy Annotation, Label Correction, Mutual Consistency, Parzen Window Estimation, Non-Convex Optimization and Spannogram Framework

1. Introduction

Labeled training data is essential for supervised classification. As annotating large datasets can be time consuming, nowadays often crowd sourcing [1] (e.g.

*Corresponding author
Email addresses: sahely@mpi-inf.mpg.de (Sahey Bhadra), hein@cs.uni-saarland.de (Matthias Hein)
URL: http://www.mpi-inf.mpg.de/~sahely/ (Sahey Bhadra), http://www.ml.uni-saarland.de/people/hein.htm (Matthias Hein)

Amazon mechanical turk) is used as a quick and cost effective solution. However, annotations acquired by crowd sourcing are generally contaminated with noise. Often it happens that some annotators do not understand the task correctly and thus provide wrong labels. Even more severe is adversarial manipulation of the training data to change the classifier in a "maximal" way. It is obvious that these different types of label noise can have a significant adverse effect on the classification performance. In the literature many negative results [2, 3, 4] have been shown regarding hardness of learning under adversarial or malicious noise.

Being able to cope with such label noise is therefore an important practical problem which has recently attracted a lot of attention. One can identify two major directions among prior work. The first one attempts to correct mislabeled examples during model building [5, 6, 7, 8, 9], while the second one applies noise filtering as a pre-processing step prior to the model building [10, 11, 12]. Typically, pre-processing techniques tend to be less prone to over-fitting as they are independent of the final classifier.

Our approach follows the second direction for label correction and involves maximization of a global consistency criterion which predicts the label of a training data point based on its neighbors using a Parzen window type approach. This, in spirit is close to the work of [10]. However, unlike their approach, we enforce hard decisions, that is either a label is wrong or right. For model selection we assume that a small fraction of the training annotations has been provided by an expert for which we assume that all the labels are correct.

Similar to other approaches for label noise correction [10, 5, 6, 7], the resulting optimization approach is non-convex. In our case, it is an NP-hard combinatorial problem. However, it turns out that in the particular setting we are working, we can develop an algorithm based on the Spannogram technique [13, 14], for which we can provide quite tight approximation guarantees. Moreover, we show in the experiments that our optimization technique outperforms standard methods based on sequential linearization, which are often employed in machine learning. Thus we think that, modifications of the Spannogram

technique could also be of potential interest in other areas of machine learning which deal with some other combinatorial problem.

We show on a large number of datasets with different types of noise that the proposed method is able to detect noisy labels with high precision and recall. This even holds up to a point where more than 40% of the training data-points are noisy. Finally, we show that, two different classifiers (SVM and Parzen window) trained using data pre-processed by our technique outperform or atleast performs similarly to the classifiers trained using data pre-processed by other existing thechniques and also their robust counterpart.

2. Related Work

Most of the existing work for detecting noisy labels as pre-processing techniques such as [2, 15, 12] uses some kind of local learning. They learn a few local classifiers from sub-sampled training data and then try to detect the correct label of a training data point via a majority vote among the local classifiers. The main problem of such greedy approaches is that they examine each training data point individually without using the mislabeling information of the sub-sampled set. When the observed labels of a large portion of the sub-sampled set are noisy then the local classifiers based on them can be completely wrong and will in the worst case insert even more noise.

Given the training dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ such that , $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ and $y_i \in \mathcal{Y} = \{1, -1\}$ a global approach to the problem has been put forward by [10], where they assume that each labeled data point x_i is noisy with an unknown probability $p_i \in [0, 1]$. Hence the expected class label of the i^{th} data point is $E[y_i] = (1 - p_i)y_i + p_i(-y_i) = (1 - 2p_i)y_i$. Given a non-negative kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$, the predicted value of $\mathbf{x}_i$ based on a Parzen window type classifier is

$$f(\mathbf{x}_i) = \frac{\sum_{j=1}^n (1 - 2p_j) y_j K(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{j=1}^n K(\mathbf{x}_i, \mathbf{x}_j)}.$$

Finally, [10] suggests a criterion to find p_i in order to maximize the label consistency with respect to the expected labels i.e $\sum_i E[y_i] f(\mathbf{x}_i)$ for the whole

dataset. This leads to the following optimization problem

$$\max_{0\leq p_i\leq 1} \langle (1-2\mathbf{p}), Q^{norm}(1-2\mathbf{p})\rangle - C\|\mathbf{p}\|_1 \tag{1}$$

where $Q^{norm}_{ij} = \frac{y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)}{\sum_j K_{(\mathbf{x}_i,\mathbf{x}_j)}}$. The regularization term $\|\mathbf{p}\|_1$ is added in order to enforce sparsity in the solution. Moreover, for a non-negative kernel function without regularization term the optimization problem will result into a trivial solution where

$$p_i = \begin{cases} 0 & if\ y_i = 1 \\ 1 & otherwise \end{cases} \quad \text{or} \quad p_i = \begin{cases} 1 & if\ y_i = 1 \\ 0 & otherwise \end{cases}$$

Note that, Q^{norm} is not necessarily a negative semi definite matrix and hence (1) is a non-convex problem.

Notation. $\|.\|_p$= l_p norm, $|\mathbf{I}|$ =cardinality of set $\mathbf{I}$, the eigenvalues λ_j of Q with corresponding eigenvector $\mathbf{v}_j$ are in decreasing order, $\mathbf{1}$ is the vector of all ones, $\lfloor z \rfloor =$ the largest integer $\leq z$, $\langle \mathbf{x}, \mathbf{v}\rangle = \sum_i x_i v_i$ and e_i is a vector with 1 in the i^{th} position and 0 else.

3. Label Noise Detection by Mutual Consistency Check

Given a training dataset $\hat{\mathcal{D}} = \{\mathbf{x}_i, y_i^t\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i^t \in \{-1, 1\}$ is the true label of the i-th data point, the Parzen window classifier [16], f^t is defined as

$$f^t(\mathbf{x}_i) = \frac{\sum_{j=1}^n y_j^t K_h(\mathbf{x}_i, \mathbf{x}_j)}{\sum_j K_h(\mathbf{x}_i, \mathbf{x}_j)}.$$

Typically, K_h is chosen to be the Gaussian kernel, $K_h(x, y) = \exp(-\frac{\|x-y\|_2^2}{2h})$, which is non-negative and positive semi-definite. Here h is the bandwidth of the kernel. Note that f^t takes values in $[-1, 1]$. Hence the loss in terms of true labels can be defined as $L(y_i^t, f^t(\mathbf{x}_i)) = 1 - y_i^t f^t(\mathbf{x}_i)$. Finally, we use a weighted loss over $\hat{\mathcal{D}}$ where the idea is that we penalize errors more in regions of high density (where we have a lot of nearby points and thus can be more sure that

the simple Parzen window classifier is correct) than in regions of low density,

$$\text{Loss}(f^t, \hat{\mathcal{D}}) \quad = \quad \frac{1}{n}\sum_{i=1}^{n} w_i L(y_i^t, f^t(\mathbf{x}_i)), \tag{2}$$

where the weight is $w_i = \frac{1}{n}\sum_{j=1}^{n} K_h(\mathbf{x}_i, \mathbf{x}_j)$. After proper rescaling of w_i, it is a consistent[1] density estimator [17], i.e., $\frac{w_i}{h^d} = \frac{1}{nh^d}\sum_{j=1}^{n} K_h(\mathbf{x}_i, \mathbf{x}_j)$ is a consistent density estimator: $\frac{w_i}{h^d} \to p(x_i)$ if $h \to 0,\ n \to \infty,$ and $nh^d \to \infty$.

In this paper we assume that we do not know the true labels and the observed labels may be noisy in the sense that some of the given labels are different from the true labels. More precisely, given noisy training data $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ we assume that the annotation y_i is a perturbed version of the true label y_i^t. For binary classification problems, the most intuitive noise model is $y_i = \eta_i y_i^t$, where $\eta_i \in \{1, -1\}$. $\eta_i = 1$ indicates a correctly observed label and $\eta_i = -1$ indicates a noisy label. The goal is to find y^t and hence η by minimizing the global loss in (2) with respect to η. We have

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n} w_i L(y_i^t, f^t(\mathbf{x}_i)) \quad &= \quad \frac{1}{n^2}\sum_{i,j=1}^{n} K_h(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{n^2}\sum_{i,j=1}^{n} y_i^t y_j^t K_h(\mathbf{x}_i, \mathbf{x}_j) \\ &= \quad \frac{1}{n^2}\sum_{i,j=1}^{n} K_h(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{n^2}\sum_{i,j=1}^{n} \eta_i y_i \eta_j y_j K_h(\mathbf{x}_i, \mathbf{x}_j) \end{aligned} \tag{3}$$

Here $\frac{1}{n^2}\sum_{i,j=1}^{n} K_h(\mathbf{x}_i, \mathbf{x}_j)$ is constant. Hence in order to maximize mutual consistency in the labels of all training data-points, we have to minimize the loss (3) and hence maximize the following optimization problem

$$\eta^* \quad = \quad \underset{\eta \in \{-1,1\}^n}{\arg\max} \sum_{i,j=1}^{n} \eta_i \eta_j y_i y_j K_h(\mathbf{x}_i, \mathbf{x}_j). \tag{4}$$

Unlike greedy approaches, above optimization problem detects all the noisy points simultaneously.

Note that, the optimization problem (4) has a trivial solution $\eta_i = y_i$ and hence is of no interest. The problem becomes challenging as we have two kinds of extra information on the problem and we encode them as constraints. First,

[1]Note that, consistent has been mentioned earlier in a different sense.

we make the natural assumption that a small part of the data set is annotated by experts and the corresponding labels are always correct. Thus the labels given by experts are fixed during the optimization. Let us define $\mathbf{I}_E$ as the set of indices corresponding to data points annotated by expert annotators. Then $\eta_i = 1,\ \forall i \in \mathbf{I}_E$. The second kind of extra information are the fractions of data points with noisy labels given by ρ_+ and ρ_- for positive and negative annotated classes respectively. Hence $\rho_+ = \frac{|\{i|y_i=1\ and\ y_i^t=-1\}|}{n_+}$ and $\rho_- = \frac{|\{i|y_i=-1\ and\ y_i^t=1\}|}{n_-}$ where $n_+ = |I_+| = |\{i|y_i = 1\}|$ and $n_- = |I_-| = |\{i|y_i = -1\}|$. The fractions of noisy labels ρ_+ and ρ_- can be estimated similar to [7], but in this paper we estimate them by cross-validation using the knowledge of the expert labels. The final constraint set of our problem (4) is defined as

$$\mathcal{E} = \{\eta \mid \eta \in \{1,-1\}^n,\ \left\langle \mathbf{1}, \eta_{\mathbf{I}_+} \right\rangle = n_+ - 2\delta_+,\ \left\langle \mathbf{1}, \eta_{\mathbf{I}_-} \right\rangle = n_- - 2\delta_-, \eta_{\mathbf{I}_E} = 1\} \quad (5)$$

where $\delta_+ = \lfloor \rho_+ n_+ \rfloor$ and $\delta_- = \lfloor \rho_- n_- \rfloor$. Hence our Label Noise Detection (**LND**) method solves the following optimization problem.

$$\textbf{LND}: \qquad \eta^* \quad = \quad \arg\max_{\eta \in \mathcal{E}} \langle \eta, Q\eta \rangle \qquad (6)$$

where, $Q_{ij} = y_i y_j K_h(\mathbf{x}_i, \mathbf{x}_j)$. Along with the constraint (5) the solution of (6) is no more trivial and we present an efficient algorithm to solve this problem in Section 4.2.

4. Algorithms

The optimization problem **LND** is a combinatorial problem and NP-hard in general. We present two possible ways to solve it approximately. The first one linearizes the objective function in each step and then solves the corresponding constrained maximization problem in closed form and the second one is an algorithm using the Spannogram technique for which we provide approximation guarantees.

4.1. Sequential Linearization

Algorithm 1 linearizes the objective function in the k-th step as $\left\langle \eta, Q\eta^k \right\rangle$ and then solves the corresponding maximization problem, $\max_{\eta \in \mathcal{E}} \left\langle \eta, Q\eta^k \right\rangle$. It

turns out that the solution is a simple projection onto the constraint set $\mathcal{E}$. **LND** solved using Algorithm 1 is denoted as $\mathbf{LND}_{slp}$.

Lemma 1. $\arg\max_{\eta \in \mathcal{E}} \langle \eta, \mathbf{v} \rangle \equiv \Pi_{\mathcal{E}}(\mathbf{v})$ *where* $\Pi_{\mathcal{E}}$ *is projection on* $\mathcal{E}$.

Proof. $\forall \eta \in \mathcal{E}$, $\|\eta\|^2 = n$ is constant and hence

$$\Pi_{\mathcal{E}}(\mathbf{v}) = \arg\min_{\eta \in \mathcal{E}} \|\eta - \mathbf{v}\|^2 \equiv \arg\max_{\eta \in \mathcal{E}} \langle \eta, \mathbf{v} \rangle .$$

□

Moreover, it turns out that the projection onto the discrete set $\mathcal{E}$ can be easily computed in closed form using Algorihtm 2 .

Theorem 1. *Algorithm 2 computes* $\Pi_{\mathcal{E}}(\mathbf{v})$.

Proof. We give a proof by contradiction. Let us assume that η, the outcome from Algorithm 2, is not equal to $\Pi_{\mathcal{E}}(\mathbf{v})$ and there exists an $\eta^* \neq \eta$ such that $\eta^* = \Pi_{\mathcal{E}}(\mathbf{v})$. Hence there are at-least two indices (j, l) such that $y_j = y_l$ but $\eta^*_j = -\eta_j$, $\eta^*_l = -\eta_l$ and $\eta_l = -\eta_j$. Without loss of generality let us assume that $v_l \geq v_j$ and hence according to Algorithm 2 $\eta_l = 1$, $\eta_j = -1$ and hence according to above assumption $\eta^*_l = -1$ and $\eta^*_j = 1$. Assuming $\eta^*_i = \eta_i \;\; \forall i \neq j, l$, we get

$$\|\eta - \mathbf{v}\|^2 - \|\eta^* - \mathbf{v}\|^2 = -2(\eta_l v_l + \eta_j v_j) + 2(\eta^*_l v_l + \eta^*_j v_j) = 4(v_j - v_l).$$

Now, as $v_l \geq v_j$, $4(v_j - v_l) \leq 0$ or $\|\eta - \mathbf{v}\|^2 \leq \|\eta^* - \mathbf{v}\|$. Hence η must be equal to $\Pi_{\mathcal{E}}(\mathbf{v})$. □

Finally, we can show that Algorithm 1 leads to monotonic ascent.

Lemma 2. *Algorithm 1 at each step provides a feasible* η^{k+1} *with monotonically increasing function value if* K_h *is positive definite.*

Proof. If K_h is positive definite, then Q is also positive definite. Using the first-order condition of convex functions, we get $\langle \eta, Q\eta \rangle \geq \langle \eta^k, Q\eta^k \rangle + \frac{1}{2} \langle Q\eta^k, \eta - \eta^k \rangle$. Thus the algorithmic scheme maximizes a lower bound on the objective. Moreover, as η^k is feasible, we get

$$\langle \eta^{k+1}, Q\eta^{k+1} \rangle \geq \langle \eta^k, Q\eta^k \rangle + \frac{1}{2} \langle Q\eta^k, \eta^{k+1} - \eta^k \rangle \geq \langle \eta^k, Q\eta^k \rangle .$$

Algorithm 1 LND$_{slp}$

Initialization: Randomly take η^0 in $\mathcal{E}$ and k := 0.

Output: η^k

repeat

Iteration k: $\eta^{k+1} = \Pi_{\mathcal{E}}(Q\eta^k)$.

until $\frac{\langle \eta^{k+1}, Q\eta^{k+1}\rangle - \langle \eta^k, Q\eta^k\rangle}{\langle \eta^k, Q\eta^k\rangle} \leq \epsilon$

Algorithm 2 $\Pi_{\mathcal{E}}(\mathbf{v})$

Initialization: $\eta = \mathbf{1}$ and $I_\eta = \emptyset$.

Output: η

$I_{\eta_+} \leftarrow \{i | \mathbf{v}_{i\in\{i|y_i=1\}} \leq \mathbf{v}_+^{[\delta_+]}$, where $\mathbf{v}_+^{[\delta_+]} = \delta_+$-th smallest element of $\mathbf{v}_{i\in\{i|y_i=1\}}\}$

$I_{\eta_-} \leftarrow \{i | \mathbf{v}_{i\in\{i|y_i=-1\}} \leq \mathbf{v}_-^{[\delta_-]}$, where $\mathbf{v}_-^{[\delta_-]} = \delta_-$-th smallest element of $\mathbf{v}_{i\in\{i|y_i=-1\}}\}$

$I_\eta = \{I_{\eta_+}, I_{\eta_-}\}$

$\eta_{I_\eta} = -1$

□

4.2. Algorithm based on low rank approximation

The **LND**$_{slp}$ (Algorithm 1) has the problem that it can get stuck in local optima without any approximation guarantees. On the other-hand, finding the global optimum of (6) is NP-hard [18]. In this paper we propose an algorithm based on the Spannogram framework [13, 14] which allows to solve **LND** with a certain approximation guarantee.

Efficient use of $\mathbf{I}_E$: Using ($\eta_i = 1,\ \forall i \in I_E$),

$$\langle \eta, Q\eta\rangle = \sum_{i,j\in I_E} Q_{ij} + 2 \sum_{i\notin I_E, j\in I_E} \eta_i Q_{ij} + \sum_{i,j\notin I_E} \eta_i\eta_j Q_{ij}.$$

Hence (6) can be solved by solving

$$\eta^{s*} \quad = \quad arg\max_{\eta\in\mathcal{E}_s} \langle \eta_s, Q_s\eta_s\rangle \tag{7}$$

where, $Q_s = \begin{bmatrix} \langle \mathbf{1}, Q_{i\in I_E \;\; j\in I_E}\mathbf{1}\rangle & \langle \mathbf{1}, Q_{i\in I_E \;\; j\notin I_E}\rangle \\ Q_{i\notin I_E \;\; j\in I_E}\mathbf{1} & Q_{i\notin I_E \;\; j\notin I_E} \end{bmatrix}$ and the new feasibility set is defined as, $\mathcal{E}_s = \{\eta | \eta_i \in \{+1, -1\},\ |\{i|\eta_i = -1 \text{ and } y_i = 1\}| = \delta_+,\ |\{i|\eta_i = -1 \text{ and } y_i = -1\}| = \delta_-$ and $\eta_1 = 1\}$. Finally we can get back the solution of (6) by assigning $\eta^*_{i\in I_E} = \eta^{s*}_1 = 1$ and $\eta^*_{i\notin I_E} = \eta^{s*}_{j\geq 2}$. As structure of problem (7) is the same as that of (6), henceforth we will consider $Q = Q_s$ and $\mathcal{E} = \mathcal{E}_s$.

For a positive semi-definite(PSD) kernel K_h, Q is also PSD hence using eigenvalue decomposition, (6) is equivalent to

$$\arg\max_{\eta\in\mathcal{E}} \langle \eta, Q_n\eta\rangle, \qquad \text{where} \quad Q_n = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \mathbf{v}_j^T \tag{8}$$

where n is number of training data points. A low rank approximation of (6) and (8) is given by

$$\arg\max_{\eta\in\mathcal{E}} \langle \eta, Q_r\eta\rangle, \qquad \text{where} \quad Q_r = \sum_{j=1}^{r} \lambda_j \mathbf{v}_j \mathbf{v}_j^T \tag{9}$$

where Q_r is the low rank approximation of Q and ideally $r << n$. This in turn is equivalent to

$$\arg\max_{\eta\in\mathcal{E}} \|\mathbf{V}_r\eta\|_2^2, \text{ where } \mathbf{V}_r = [\sqrt{\lambda_1}\mathbf{v}_1, \ldots, \sqrt{\lambda_r}\mathbf{v}_r]^T.$$

Let $\mathbf{c}$ be a $r \times 1$ unit length vector, i.e., $\|\mathbf{c}\|_2 = 1$. Using Cauchy-Schwarz inequality, we get $\langle \mathbf{c}, \mathbf{V}_r\eta\rangle^2 \leq \|\mathbf{V}_r\eta\|_2^2$; with equality, if and only if, $\mathbf{c}$ is co-linear to $\mathbf{V}_r\eta$. Hence (9) is equivalent to

$$\arg\max_{\eta\in\mathcal{E}} \max_{\mathbf{c}:\|\mathbf{c}\|^2=1} \langle \mathbf{c}, \mathbf{V}_r\eta\rangle^2. \tag{10}$$

For a given $\mathbf{c}^*$, defining $\mathbf{v}_r^{c^*} = \mathbf{V}_r^T\mathbf{c}^*$, (10) is equivalent to

$$\arg\max_{\eta\in\mathcal{E}} \langle \mathbf{c}^*, \mathbf{V}_r\eta\rangle^2 = \arg\max_{\eta\in\mathcal{E}} \left|\left\langle \mathbf{v}_r^{c^*}, \eta\right\rangle\right|.$$

Lemma 3. $\arg\max_{\eta\in\mathcal{E}} |\langle \mathbf{v}_r^c, \eta\rangle| \in \mathbf{S}^{\mathbf{c}}_{(r,\mathcal{E})}$, *where* $\mathbf{S}^{\mathbf{c}}_{(r,\mathcal{E})} = \{\Pi_{\mathcal{E}}(\mathbf{v}_r^c), \Pi_{\mathcal{E}}(-\mathbf{v}_r^c)\}$.

Proof. We have, $\arg\max_{\eta\in\mathcal{E}} |\langle \mathbf{v}_r^c, \eta\rangle| \in \{\arg\max_{\eta\in\mathcal{E}} \langle \mathbf{v}_r^c, \eta\rangle,\ \arg\max_{\eta\in\mathcal{E}} \langle(-\mathbf{v}_r^c), \eta\rangle\}$. Now, using Lemma 1 we get, $\arg\max_{\eta\in\mathcal{E}} |\langle \mathbf{v}_r^c, \eta\rangle| \in \{\Pi_{\mathcal{E}}(\mathbf{v}_r^c),\ \Pi_{\mathcal{E}}(-\mathbf{v}_r^c)\} = \mathbf{S}^{\mathbf{c}}_{(r,\mathcal{E})}$. □

Now for solving (10) the remaining part is to find $\mathbf{c}^*$ such that

$$\mathbf{c}^* = arg \max_{\mathbf{c}:\|\mathbf{c}\|^2=1} \max_{\eta \in \mathbf{S}^{\mathbf{c}}_{(r,\mathcal{E})}} \langle \mathbf{c}, \mathbf{V}_r \eta \rangle^2 . \tag{11}$$

For $r = 1$, that is $c \in \mathbb{R}$, there are only two feasible $\mathbf{c}$ such that $\|\mathbf{c}\|^2 = 1$ and they are given by $\mathbf{c} = \pm 1$ and hence instead of solving (11) one can directly solve (10) by

$$\arg\max_{\eta \in \mathcal{E}} \max_{\mathbf{v}^c_r \in \pm\sqrt{\lambda_1}\mathbf{v}_1^T} \langle \mathbf{v}^c_r, \eta \rangle^2 .$$

Similarly, for $r \geq 2$ instead of solving (11) from an infinitely large set of feasible $\mathbf{c}$ we find a finite set of potential $\mathbf{c}$ denoted as $\mathcal{C}_r$ such that

$$\arg\max_{\eta \in \mathcal{E}} \max_{\mathbf{c}:\|\mathbf{c}\|^2=1} \langle \mathbf{c}, \mathbf{V}_r \eta \rangle^2 \in \cup_{\mathbf{c} \in \mathcal{C}_r} \arg\max_{\eta \in \mathbf{S}^{\mathbf{c}}_{(r,\mathcal{E})}} \langle \mathbf{c}, \mathbf{V}_r \eta \rangle^2 .$$

Please note that for a fixed $\mathbf{c}$, $\arg\max_{\eta \in \mathcal{E}} \langle \mathbf{c}, \mathbf{V}_r \eta \rangle^2$ is solved using $\Pi_{\mathcal{E}}$ (Lemma 3) where $\Pi_{\mathcal{E}}$ (or Algorithm 2) uses the sorted order of elements of $\mathbf{v}^c_r$. Again an $\eta \in \mathcal{E}$ can be a potential solution of (9) only if there exist a $\mathbf{c}$ for which $\eta \in \Pi_{\mathcal{E}}(\mathbf{v}^c_r)$. Hence it is enough to build $\mathcal{C}_r$ which contains all $\mathbf{c}$ which generate a different sorted order of elements of $\mathbf{v}^c_r$.

4.2.1. The Spannogram framework

The key idea of the Spannogram framework [13, 14] is the introduction of spherical coordinates. For any $r \geq 2$ this transformation can be done by using $r-1$ phase variable $\Phi = [\phi_1, \ldots, \phi_{r-1}] \in [[-\frac{\pi}{2}, \frac{\pi}{2}]^{(r-2)}, [-\pi, \pi]]$ by expressing $\mathbf{c}$ without loss of generality as

$$\mathbf{c} = \begin{bmatrix} \sin(\phi_1) \\ \cos(\phi_1)\sin(\phi_2) \\ \cos(\phi_1)\cos(\phi_2)\sin(\phi_3) \\ \ldots \\ \cos(\phi_1)\cos(\phi_2)\ldots\sin(\phi_{r-1}) \\ \cos(\phi_1)\cos(\phi_2)\ldots\cos(\phi_{r-1}) \end{bmatrix} \tag{12}$$

which is a vector of unit norm and for all ϕ it produces all $r \times 1$ unit vectors. Under this transformation $\mathbf{v}^c_r$ can be expressed in terms of ϕ as

$$\begin{aligned}\mathbf{v}(\Phi) &= \sin(\phi_1)[\sqrt{\lambda_1}\mathbf{v}_1] + \cos(\phi_1)\sin(\phi_2)[\sqrt{\lambda_2}\mathbf{v}_2] + \dots \\ &\quad + \cos(\phi_1)\cos(\phi_2)\dots\cos(\phi_{r-1})[\sqrt{\lambda_r}\mathbf{v}_r] \end{aligned} \tag{13}$$

where each element of $[\mathbf{v}(\Phi)]_i$ is continuous function of $r-1$ variables Φ. Calculating $\Pi_{\mathcal{E}}(\mathbf{v}_r^c)$ for a fixed vector $\mathbf{c}$ is equivalent to finding the relative sorting of the n surfaces $[\mathbf{v}(\Phi)]_{i=1,\dots,n}$ for corresponding Φ. If the relative ranking of i-th and j-th elements of $\mathbf{v}(\Phi_1)$ and $\mathbf{v}(\Phi_2)$ change, i.e, $[\mathbf{v}(\Phi_1)]_i > [\mathbf{v}(\Phi_1)]_j$ but $[\mathbf{v}(\Phi_2)]_i < [\mathbf{v}(\Phi_2)]_j$, then there must be a $\Phi_3 \in [\Phi_1, \Phi_2]$ such that $[\mathbf{v}(\Phi_3)]_i = [\mathbf{v}(\Phi_3)]_j$. Hence it is enough to collect all the points where any two of these surfaces intersect. Please note that, the solution of $[\mathbf{v}(\Phi)]_i = [\mathbf{v}(\Phi)]_j$ is not a single point (i.e., a single vector $\mathbf{c}$) but a $(r-1)$ dimensional space of solutions denoted as

$$\mathbf{\Phi}_{i,j} = \{\phi \mid [\mathbf{v}(\phi)]_i = [\mathbf{v}(\phi)]_j\}.$$

Since $\Pi_{\mathcal{E}}$ and hence local optimum changes only if the local ranking changes, the intersection points defined by all $\mathbf{\Phi}_{i,j}$ sets are the only points of interest. For the vectors in this space $\mathbf{\Phi}_{i,j}$, there are again some critical ϕ's where both the i-th and j-th elements are greater than δ_+-th (δ_--th) largest or less than δ_+-th (δ_--th) smallest element of $\mathbf{v}(\phi)$ and still sorted order of element of $\mathbf{v}(\phi)$ changes at ϕ. This happens when both i-th and j-th elements of $\mathbf{v}(\phi)$ become equal to the l-th element of $\mathbf{v}(\phi)$. This new $(r-3)$ dimensional sub-space is denoted as $\mathbf{\Phi}_{i,j,l}$. At this point, the intersection points defined by all $\mathbf{\Phi}_{i,j,l}$ sets are the only points of interest. In this manner we can find $\mathbf{\Phi}_{i_1,i_2,\dots,i_r}$ which contains all Φ such that

$$[\mathbf{v}(\Phi)]_{i_1} = [\mathbf{v}(\Phi)]_{i_2} = \dots = [\mathbf{v}(\Phi)]_{i_r}.$$

Finally, the intersection points defined by all such $\mathbf{\Phi}_{i_1,i_2,\dots,i_r}$ sets are the only points of interest. Hence only $\mathbf{c}$ corresponding to $\phi \in \mathbf{\Phi}_{i_1,i_2,\dots,i_r}$ need to be checked and can be obtained by solving the system of $r-1$, linear equations

Algorithm 3 Computing $\mathcal{C}_r$ corresponding to $\mathbf{V}_r$

Initialization: $\mathcal{C}_r = \emptyset$

Output: $\mathcal{C}_r$

for all class y **do**

for all $\binom{n_y}{r}$ subsets $\{i_1, \ldots, i_r\} \subset \{i | i \in \mathbf{I}_y / \mathbf{I}_E\}$ **do**

$$\mathbf{c} = nullspace\left(\begin{bmatrix} e_{i_1}^T - e_{i_2}^T \\ \cdots \\ e_{i_1}^T - e_{i_r}^T \end{bmatrix} \mathbf{V}_r^T\right)$$

sort elements of $\mathbf{v}_r^c = \mathbf{V}_r^T \mathbf{c}$

if $\exists (i \in \{i_1, \ldots, i_r\})$ such that $\mathbf{v}_{r_i}^c$(or $-\mathbf{v}_{r_i}^c$) is equal to δ_y -th smallest element of $\mathbf{v}_r^c$(or $-\mathbf{v}_r^c$) **then**

$\mathcal{C}_r \leftarrow \mathcal{C}_r \cup \{\mathbf{c}\}$

end if

end for

end for

$$\begin{bmatrix} e_{i_1}^T - e_{i_2}^T \\ \cdots \\ e_{i_1}^T - e_{i_r}^T \end{bmatrix} \mathbf{v}(\Phi) = \begin{bmatrix} e_{i_1}^T - e_{i_2}^T \\ \cdots \\ e_{i_1}^T - e_{i_r}^T \end{bmatrix} \mathbf{v}_r^c = \begin{bmatrix} e_{i_1}^T - e_{i_2}^T \\ \cdots \\ e_{i_1}^T - e_{i_r}^T \end{bmatrix} \mathbf{V}_r^T \mathbf{c} = 0. \tag{14}$$

where e_i is a vector of all zeros except 1 in the i-th position.

Please note that, we need only the sorted order of elements corresponding to each annotation separately. Algorithm 3 finds such $\mathbf{c}$ considering elements of $\mathbf{v}_r^c$ corresponding to all positively and negatively annotated data points separately. Again, we will consider those $\mathbf{c}$ for which at least one of these r elements of $\mathbf{v}_r^c$ with equal values is equal to δ_+-th (δ_--th) smallest (or largest) element of $\mathbf{v}_r^c$.

Finally, **LND** solved by Algorithm 4 is denoted as $\mathbf{LND}_r$ for r-rank approximation.

Similarity with other algorithms: The Spannogram framework is inspired by the work of [13, 14], where sparse PCA and the densest subgraph problem have been studied. Both of these problems are quadratic maximization prob-

Algorithm 4 LND$_r$

Input: $\mathcal{E}$, Q, r

Output: η_r^*

Compute $\mathbf{V}_r = [\sqrt{\lambda_1}\mathbf{v}_1, \ldots, \sqrt{\lambda_r}\mathbf{v}_r]^T$.

Build $\mathcal{C}_r$ (Algorithm 3)

Build $\mathbf{S}_{(r,\mathcal{E})} = \cup_{\mathbf{c}\in\mathcal{C}_r}\mathbf{S}^{\mathbf{c}}_{(r,\mathcal{E})}$ (Lemma 3)

$\eta_r^* = arg\max_{\eta\in\mathbf{S}_{(r,\mathcal{E})}} \langle \eta, Q\eta\rangle$

lems, like **LND** in (6). Unlike **LND** they need sparse solutions while **LND** has a $\{1,-1\}$ constraint on the variables which requires a fundamental modification in the theoretical analysis. [14] seems to be the closest to the proposed algorithm where the Spannogram technique is used to maximize a quadratic function with $\{0,1\}$ constraint. For projecting onto the cardinality constraint [14] needs a sorted order of elements of a vector similar to $\mathbf{v}_r^c$ and to get different possible rankings they use the Spannogram technique. For our problem the projection onto $\mathcal{E}$ also has a closed form solution depending on the sorted order of elements. This similarity between both the problems motivate us to extend the Spannogram framework for solving **LND**. We show that the Spannogram type algorithms can also be used when the required solution is not sparse. The proposed algorithm can also be easily extended to solve similar problems with other integer constraints on η in place of $\{1,-1\}$ constraint by assigning $\eta_i \geq \eta_j$ when $\mathbf{v}^c_{r_i} > \mathbf{v}^c_{r_j}$ and again $\eta_i \geq \eta_j$ when $-\mathbf{v}^c_{r_i} > -\mathbf{v}^c_{r_j}$.

Complexity of proposed algorithm: For a rank-r approximation we have to solve a set of $\binom{n_+}{r}$ and $\binom{n_-}{r}$ equations to find $\mathcal{C}_r$. Each of these equation sets will add one $\mathbf{c}$ in $\mathcal{C}_r$ and add at-most $2\binom{r}{\frac{r}{2}}$ candidates in $\mathbf{S}^c_{(r,\mathcal{E})}$. Hence, $|\mathbf{S}_{(r,\mathcal{E})}|$ is less than $2\binom{r}{\frac{r}{2}}\left(\binom{n_+}{r}+\binom{n_-}{r}\right)$ or $O\left(\binom{\max\{n_+,n_-\}}{r}\right)$. Considering the complexity of sorting $O(n\log n)$ and of final matrix multiplication $\langle \eta, Q\eta\rangle$ for every $\eta \in \mathbf{S}_{r,\mathcal{E}}$, the time complexity of the proposed algorithm is $O(\max\{n_+,n_-\}^r(n^2+n\log n))$. For our experiment we have used mostly $r \leq 2$ so that the search space has been reduced to $O(n^2)$ and time complexity is $O(n^4)$.

5. Approximation Guarantees

We have the following different problems,

$$\begin{aligned} OPT^* &= \max_{\eta\in\mathcal{E}} \langle \eta, Q\eta\rangle\,, & \eta^* &= arg\max_{\eta\in\mathcal{E}} \langle \eta, Q\eta\rangle\,, \\ OPT_r^* &= \max_{\eta\in\mathbf{S}_{(r,\mathcal{E})}} \langle \eta, Q\eta\rangle\,, & \eta_r^* &= arg\max_{\eta\in\mathbf{S}_{(r,\mathcal{E})}} \langle \eta, Q\eta\rangle\,, \end{aligned}$$

Here $\mathbf{S}_{(r,\mathcal{E})} = \cup_{\mathbf{c}\in\mathcal{C}_r}\mathbf{S}^{\mathbf{c}}_{(r,\mathcal{E})}$.

Lemma 4. *If $O_g = \langle \eta_g, Q\eta_g\rangle$, where $\eta_g = \Pi_{\mathcal{E}}(q_1)$ and q_1 denotes the first row of Q. Then*

$$OPT^* \;\geq\; \max\left\{O_g, \lambda_1 \langle \Pi_{\mathcal{E}}(\mathbf{v}_1), \mathbf{v}_1\rangle^2, \lambda_1 \langle \Pi_{\mathcal{E}}(-\mathbf{v}_1), \mathbf{v}_1\rangle^2\right\}. \tag{15}$$

Proof. The first part follows using $OPT^* = \max_{\eta\in\mathcal{E}} \langle \eta, Q\eta\rangle \geq \langle \eta_g, Q\eta_g\rangle$, while the second and third parts inside the max in (15) can be proven as follows.

$$\begin{aligned} OPT^* &\geq \max_{\eta\in\mathcal{E}} \langle \eta, Q_1\eta\rangle = \max_{\eta\in\mathcal{E}} \lambda_1\left(\langle \eta, \mathbf{v}_1\rangle^2\right) \text{ (as Q is PSD)} \\ &= \lambda_1 \max\left\{\langle \Pi_{\mathcal{E}}(\mathbf{v}_1), \mathbf{v}_1\rangle^2, \langle \Pi_{\mathcal{E}}(-\mathbf{v}_1), \mathbf{v}_1\rangle^2\right\}. \end{aligned} \tag{16}$$

□

Theorem 2. $OPT_r^* \geq (1-\epsilon_r)OPT^*$, *where*

$$\epsilon_r \leq \frac{n\left(\lambda_{r+1} - \lambda_n\right)}{\max\left\{O_g, \lambda_1 \langle \Pi_{\mathcal{E}}(\mathbf{v}_1), \mathbf{v}_1\rangle^2, \lambda_1 \langle \Pi_{\mathcal{E}}(-\mathbf{v}_1), \mathbf{v}_1\rangle^2\right\}}.$$

Proof. We decompose the quadratic form in (6) in two parts

$$\langle \eta, Q\eta\rangle = \langle \eta, Q_r\eta\rangle + \langle \eta, Q_{r^c}\eta\rangle$$

where $Q_r = \sum_{i=1}^{r} \lambda_i \mathbf{v}_i\mathbf{v}_i^T$ and $Q_{r^c} = \sum_{i=r+1}^{n} \lambda_i \mathbf{v}_i\mathbf{v}_i^T$. By defining, $\eta_r = arg\max_{\eta\in\mathbf{S}_{(r,\mathcal{E})}} \langle \eta, Q_r\eta\rangle$ and hence $OPT_r = \langle \eta_r, Q_r\eta_r\rangle = \max_{\eta\in\mathbf{S}_{(r,\mathcal{E})}} \langle \eta, Q_r\eta\rangle$,

$$\begin{aligned} OPT_r^* &\geq \langle \eta_r, Q\eta_r\rangle \text{ (as } \eta_r \in \mathbf{S}_{(r,\mathcal{E})}) = \langle \eta_r, Q_r\eta_r\rangle + \langle \eta_r, Q_{r^c}\eta_r\rangle \\ &\geq OPT_r + n \min_{\eta:\|\eta\|_2=1} \langle \eta, Q_{r^c}\eta\rangle \geq OPT_r + n\lambda_n \\ \Rightarrow\; OPT_r &\leq OPT_r^* - n\lambda_n \end{aligned} \tag{17}$$

Using $\max_{\eta\in\mathcal{E}} \langle \eta, Q_{r^c}\eta\rangle \leq n \max_{\eta:\|\eta\|_2=1} \langle \eta, Q_{r^c}\eta\rangle = n\lambda_{r+1}$ and (17) we get,

$$\begin{aligned} OPT^* &\leq \max_{\eta\in\mathcal{E}} \langle \eta, Q_r\eta\rangle + \max_{\eta\in\mathcal{E}} \langle \eta, Q_{r^c}\eta\rangle \leq OPT_r + n\lambda_{r+1} \\ &\quad (\max_{\eta\in\mathcal{E}} \langle \eta, Q_r\eta\rangle = \max_{\eta\in\mathbf{S}_{(r,\mathcal{E})}} \langle \eta, Q_r\eta\rangle \text{ as } Q_r \text{ is a rank-r matrix}) \\ \Rightarrow \quad OPT^* &\leq OPT_r^* - n\lambda_n + n\lambda_{r+1} \quad \text{(using (17))} \end{aligned} \tag{18}$$

One can write (18) as $OPT_r^* \geq \left(1 - \frac{n(\lambda_{r+1}-\lambda_n)}{OPT^*}\right) OPT^*$. Now the lower bound of OPT^* from Lemma 4 completes the proof.

□

Note that, when a significant number of the training data points are labeled by expert annotators the first row and the first column of Q will have significantly larger values than other elements of Q. This makes the largest eigenvalue (λ_1) much greater than other eigenvalues and hence ϵ_r becomes relatively small as is also shown in Section 6.

6. Experiments

This section presents our experimental setup and the results. We apply the proposed **LND** on a variety of data-sets contaminated with different kind of label noise. The objectives of our experiments are: (1) to illustrate the improvements in classification accuracy after correcting the label noise, (2) to compare the performance of **LND** against other existing methods in terms of ability to detect wrongly annotated data points, and (3) to prove superiority of the proposed Algorithm 4 for solving the non-convex optimization problem **LND** against the popular sequential linearization based algorithm $\mathbf{LND}_{slp}$.

6.1. Experimental Setup

Data-sets: To evaluate the performance of our method, we use 8 data-sets from [19] described in Table 1. All the experiments are repeated ten times on ten random partitions and then the average performance is reported.

The percentage of training data-set labeled by experts is fixed at 10% for most of our experiments. In our experiments we select all the parameters for

Table 1: Datasets

Name	# Data-set	Train-Test Split
Mushrooms	8124 x 112	$(50\%, 50\%)$
Svmguide1	7089 x 4	As in [19]
Fourclass	862 x 2	$(50\%, 50\%)$
Australian	619 x 14	$(50\%, 50\%)$
WDBC	569 x 30	$(50\%, 50\%)$
Heart	270 x 13	$(50\%, 50\%)$
Adult	32561 x 123	As in [19]
Covertype	581012 x 54	$(1\%, 9\%)$

various methods using 5-fold cross-validation and hence consider at-least 5 data points from each class for each data-set to be labeled by expert annotators. Considering the size of our smallest data-set (Heart), we fix this number at 10% for our experiments. But to study the effect of amount of available expert labels on performance of the proposed method, we repeated all experiments with less number (1% , 2% and 5%) of expert labels for larger data-sets (Mushrooms, Svmguide1 and Covertype). The rest of the training data-set, other than the portion annotated by experts, is contaminated by noise according to the following three different noise models.

Noise models: The idea behind these different noise models is that they reflect real life scenarios of label noise.

Boundary or margin noise (M): Here we try to mimic the situation where annotators are confused about the correct label of a data point if it is close to the decision boundary. Hence to simulate margin noise we first train a support vector machine classifier (**SVM**) and get the margin γ. Then we flip the labels of 60% of the data points lying inside the margin. To increase the noise level we widen the margin by changing the parameter C of the **SVM**.

Biased annotator noise (BA): Here we are simulating the situation when a fraction of annotators are biased towards some unknown classifier and hence

they annotate all the data points according to that specific classifier. To simulate this kind of noise we fix a random classifier and then change the labels of randomly chosen data points according to the outcome of that random classifier.

Adversarial noise (A): We follow the method described in [8] to insert adversarial noise by flipping labels of those data points which have maximum impact on the classifier.

To illustrate the effect of noisy labels on various classifiers, we repeat all the experiments by varying the number of mislabeled examples from 5% to 45% of the training data-set and use a **SVM** and a Parzen window **PW** classifier as the final classifiers. We also study the case where label noise is present only in one class.

Methods compared: We compare the proposed **LND** with **KBDMS1**[10]. In case of **KBDMS1**, we use two regularization terms $C_+\|p_{\mathbf{I}_+}\|_1$ and $C_-\|p_{\mathbf{I}_-}\|_1$ instead of $C\|p\|_1$ so that it can handle class conditional noise, where both C_+ and C_- are tuned in the range of $2^{\{-5:1:5\}}$. For learning a classifier from the outcome of the noisy label detection method **KBDMS1**, we flip the label of the i-th data point when $p_i > 0.5$. We also compare **LND** with some intuitive and simple approaches, **SubSVM** [12] and **SubPW** where we learn 50 local classifiers on the sub-sampled training data-set with a sample size of $\log_2 n$. We correct the label of every training data point except those annotated by the experts, using majority vote from predictions of these 50 sub learners. We keep the parameter values equal for all the 50 sub learners. We study the impact of noise correction methods against a nominal **SVM**, **PW** and robust counter parts of **SVM** such as **SVM** with class conditional cost (**CSVM**) [9] and Robust SVM (**RSVM**) [8]. Moreover, we compare the obtained classification accuracy with the results of classifiers trained with correctly annotated labels (**True-SVM** or **True-PW**) and trained with data points annotated by the expert annotators (**Expert-SVM** or **Expert-PW**).

The parameters ρ_+/ρ_- for all kinds of **LND** are chosen by cross-validation from $\{0 : 0.05 : 0.5\}$ using knowledge of expert labels. For cross-validation, the training data-set is divided in such a way that the data points labeled by expert

annotators are equally distributed over all the partitions and the validation error is calculated by considering the mis-classification error **only** for the data points labeled by experts. All other parameters like $C, C_+, C_- \in 2^{\{-10:2:10\}}$ for all **SVM** classifiers and $\mu \in \{0 : 0.1 : 0.5\}$ of **RSVM** are chosen also by 5-fold cross-validation in a similar way. The bandwidth h of the Gaussian kernel is tuned independently for all the algorithms varying in the range of $2^{\{-5:1:5\}}$. We are not able to compare the proposed method with **ROD** [6] as it is not scalable beyond only hundred data points.

We compare the performances of all the proposed algorithms $\mathbf{LND}_{slp}$, $\mathbf{LND}_r$ for $r = 1, 2$ and $\mathbf{LND}_{slp_1}$ to verify importance of the proposed Algorithm 4. $\mathbf{LND}_{slp_1}$ uses $\mathbf{LND}_{slp}$ with output from $\mathbf{LND}_1$ as the starting point. The results for $\mathbf{LND}_{slp}$ correspond to the best local maximum obtained from 100 random initializations.

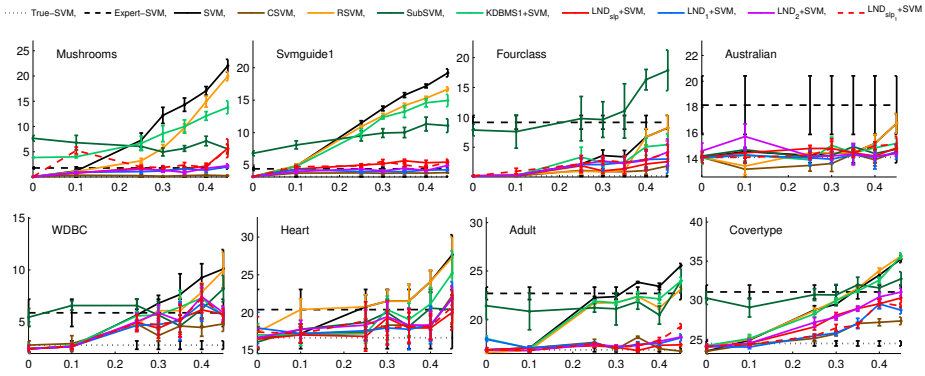
Computational setting: All experiments are done using Intel(R)-Xeon(R) (2.67GHz) processor with 36 GB RAM. For **SVM** and **SubSVM** we use LibSVM [19] with its Matlab interface. While all other algorithms are implemented with Matlab(R2013a). [2].

6.2. Results

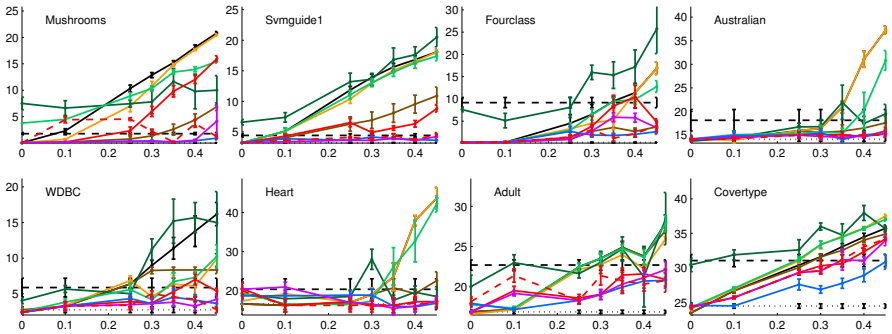
Classification accuracy: In this section we study how the correction of label noise by different algorithms influence the test error of the final classifiers (**SVM** and **PW**). Experimental results in Figure 1,2,3,4[3] indicate the fact that for both **SVM** and **PW** classifiers, correcting annotations using our model helps to get better test errors. By increasing the number of noisy labels in the training data-set test errors of **SVM** and **PW** classifiers increase heavily while after pre-processing with the proposed method ($\mathbf{LND}_1$, $\mathbf{LND}_2$ and $\mathbf{LND}_{slp_1}$) the test error increases with lower rate and in most of the cases remain very close to

[2]Both the code for LNDs and the used data-sets with label noise are available at *http://www.ml.uni-saarland.de/code/LabelNoiseCorrection/LabelNoiseCorrection.htm*

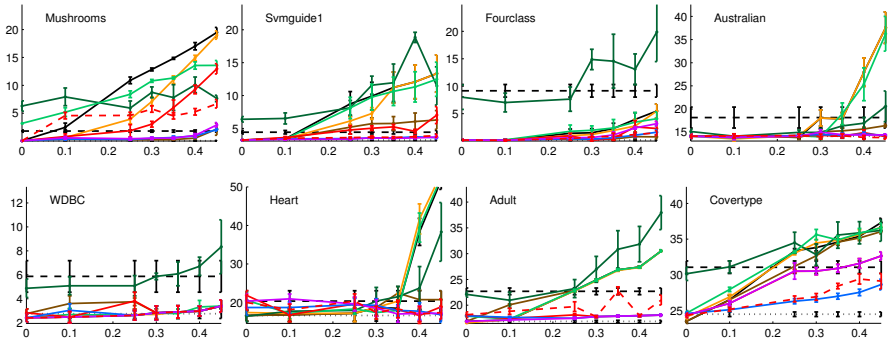
[3] Each plot shows comparison of performance of various methods on each data-set (written on the plot).



(a) Boundary Noise

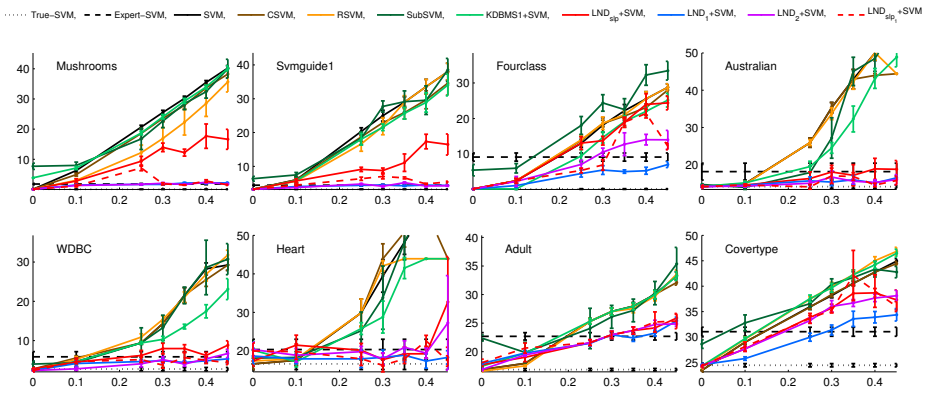


(b) Biased Annotator Noise

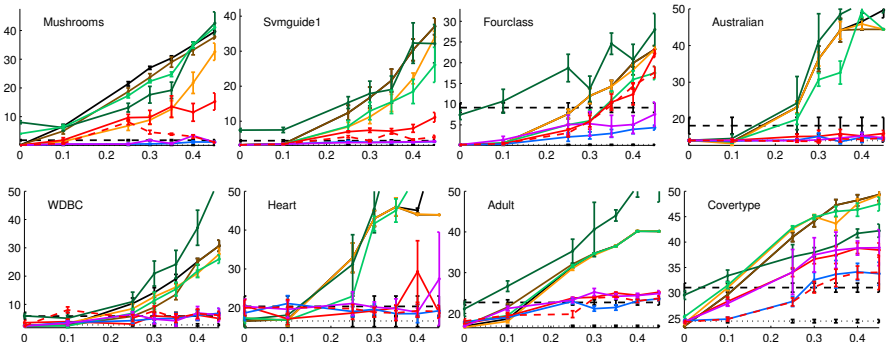


(c) Adversarial Noise

Figure 1: Comparison of different label noise correction methods and robust SVMs in terms of classification error of the **SVM** classifiers trained with corrected labels. In each plot the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in the majority class of the training data-set.



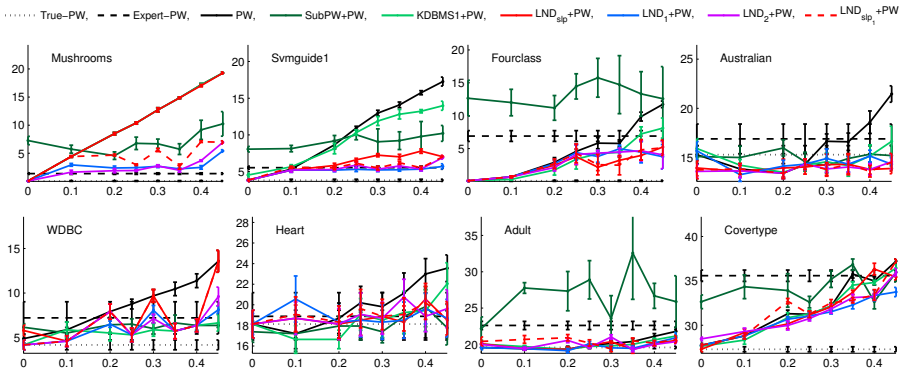
(a) Biased Annotator Noise



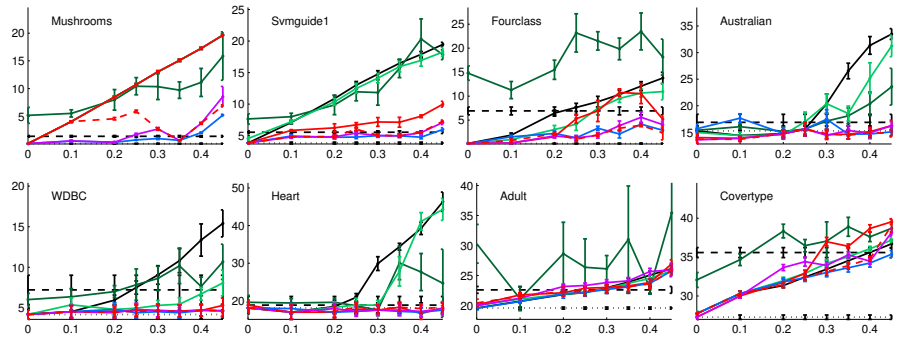
(b) Adversarial Noise

Figure 2: Comparison of different label noise correction methods and robust SVMs in terms of classification error of the **SVM** classifiers trained with corrected labels. In each plot the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in both classes of the training data-set.

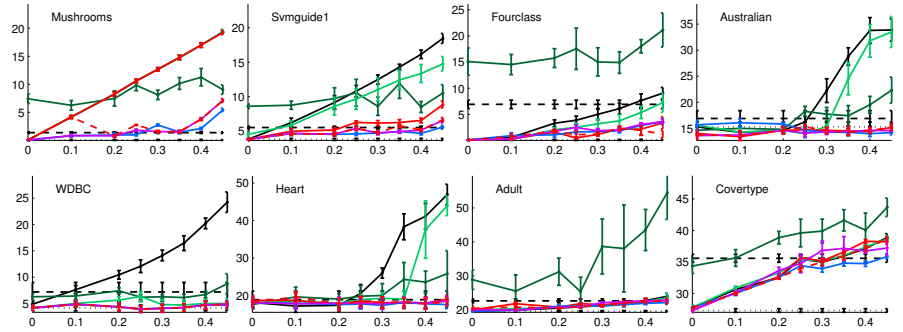
the test error of the classifier trained with true labels. **KBDMS1** performs well when the number of noisy data points is small but by increasing the number of noisy labels it deteriorates. The reason is that with a high regularization parameter it can only detect a few data points as noisy on the other hand by decreasing the value of the regularization parameters after a certain value, the effect of regularization become negligible and it starts detecting noisy data-points only from one class (as discussed in Section 2). The performance of **SubSVM** and **SubPW** are not consistent and vary for different data-sets and



(a) Boundary Noise

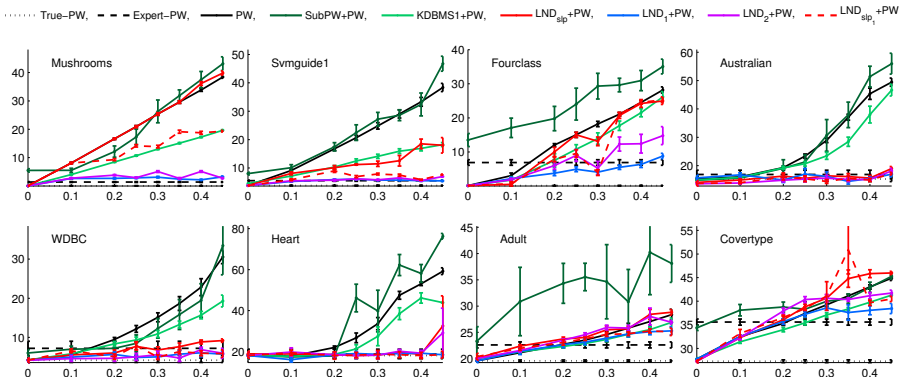


(b) Biased Annotator Noise

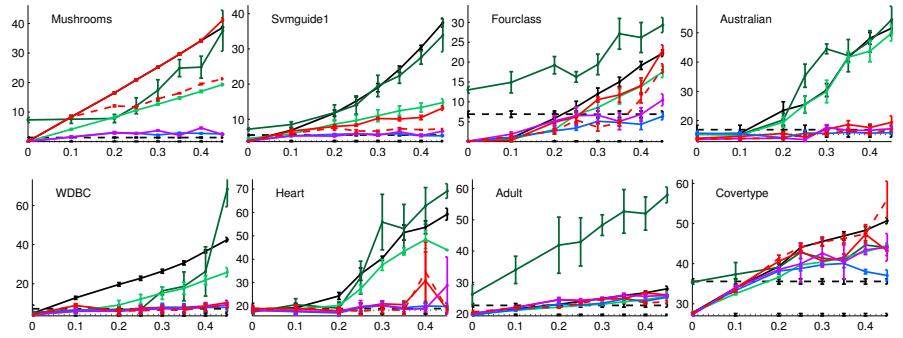


(c) Adversarial Noise

Figure 3: Comparison of different label noise correction methods in terms of classification error of **PW** classifiers trained with corrected labels. In a plot the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in the majority class of the training data-set.



(a) Biased Annotator Noise

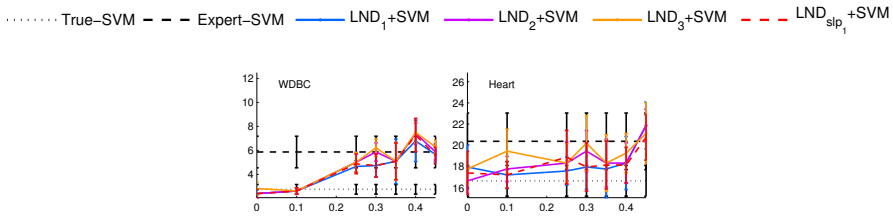


(b) Adversarial Noise

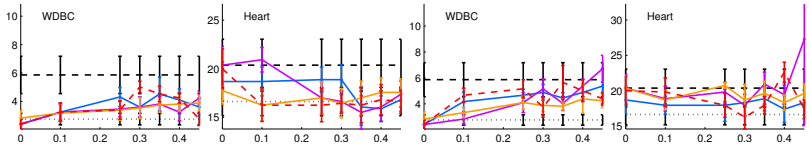
Figure 4: Comparison of different label noise correction methods in terms of classification error of **PW** classifiers trained with corrected labels. In a plot the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in both classes of the training data-set.

also have large variance.

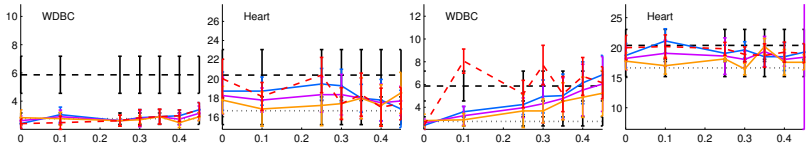
Figures 1,2 show that in general, for the biased annotator noise and adversarial noise, **SVM** classifiers trained with the labels corrected by **LND** also outperforms all the other robust counter parts (**CSVM** and **RSVM**). Only in the case of boundary noise, **CSVM** performs best but our method is close. But for other kinds of noise the performance of **CSVM** deteriorates. When the noise is symmetric for both the classes, the performance of **CSVM** is not better than the performance of **SVM** trained with noisy data. As the margin noise in both the classes does not affect **SVM** much, we have not studied that kind of



(a) Boundary Noise



(b) Biased Annotator Noise in the majority class (left) and in both classes (right)



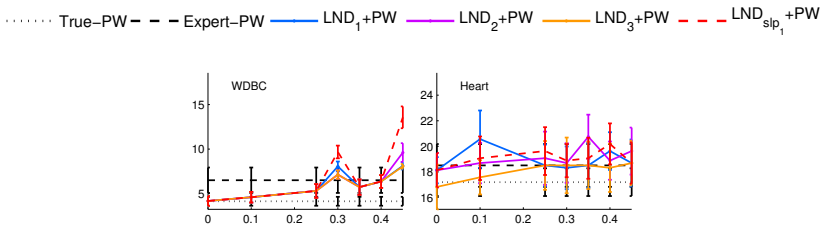
(c) Adversarial Noise in the majority class (left) and in both classes (right)

Figure 5: Effect of approximation quality for **LND** in terms of classification error of the **SVM** classifiers trained with labels corrected by $\mathbf{LND}_1$, $\mathbf{LND}_2$ and $\mathbf{LND}_3$. In the plots the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in the majority class (left) and in both classes (right) of the training data-set.

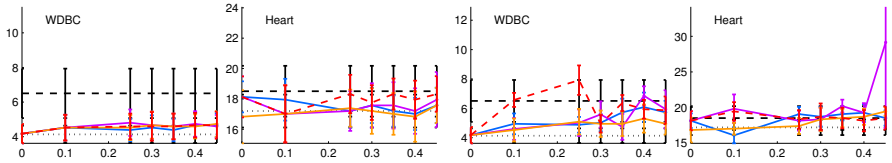
noisy data.

We also study performance of rank-3 approximation of $\mathbf{LND}(\mathbf{LND}_3)$ for the smaller data-sets: WDBC and Heart. Figure 5, 6 show that for all kinds of noise in the case of WDBC and for biased annotator and boundary noise in the case of Heart the performance of rank-1 approximation of $\mathbf{LND}(\mathbf{LND}_1)$ is also very close (sometimes better) to the performance of higher order (rank-2 and rank-3) approximations. Figure 1, 2, 3, 4 also show that one can get a good amount of improvement in terms of classification accuracy over **SVM** and **PW** classifiers by pre-processing the noisy data-sets using $(\mathbf{LND}_1)$ with a running time of $O(n^2)$ which is also feasible for large data-sets.

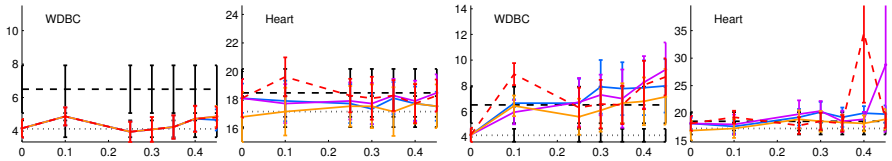
We compare classification accuracy when availability of the expert labels



(a) Boundary Noise



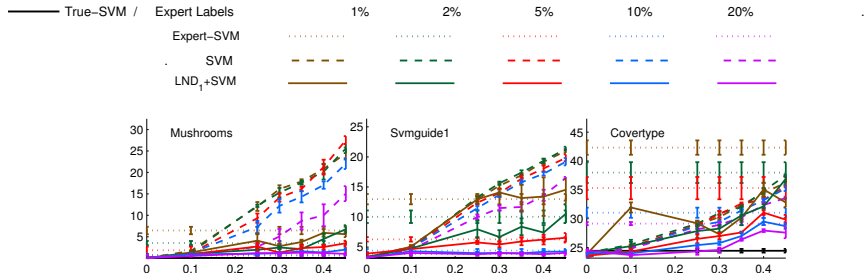
(b) Biased Annotator Noise in the majority class (left) and in both classes (right)



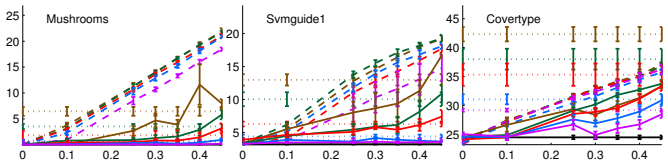
(c) Adversarial Noise in the majority class (left) and in both classes (right)

Figure 6: Effect of approximation quality for **LND** in terms of classification error of the **PW** classifiers trained with labels corrected by $\mathbf{LND}_1$, $\mathbf{LND}_2$ and $\mathbf{LND}_3$. In the plots the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in the majority class (left) and in both classes (right) of the training data-set.

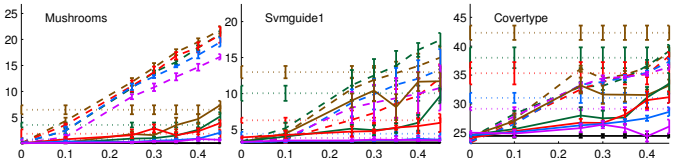
varies. Figure 7, 8 show how the performance of **Expert-SVM** (**Expert-PW**), nominal **SVM** (**PW**) and $\mathbf{LND}_1$ differ when the percentage of expert labels in the training data-set varies among 1%, 2%, 5%, 10% and 20%. Important to note that, even with 1% of expert labels, the proposed **LND** improves classification accuracy sometimes more than 10% of that achieved by nominal **SVM** and **PW**. $\mathbf{LND}_1$ also beats **Expert-SVM** and **Expert-PW** with high margin for small noise level and the difference is more when small number of expert labels are available. Please note that, we are using expert label information to tune ρ_- and ρ_+ in cross-validation. Hence, we are not able to tune these parameters when there are no expert label available. That is why, we do not include the



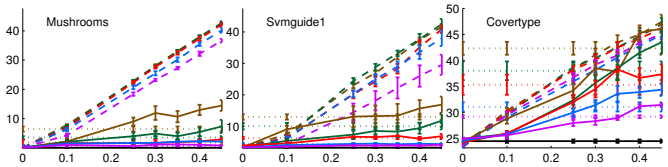
(a) Boundary Noise in the majority class



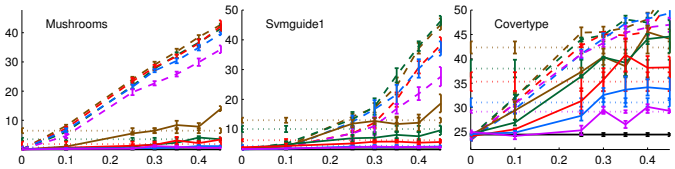
(b) Biased Annotator Noise in the majority class



(c) Adversarial Noise in the majority class

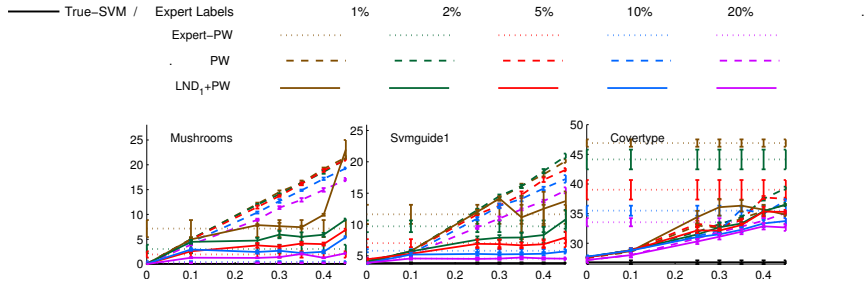


(d) Biased Annotator Noise in both classes

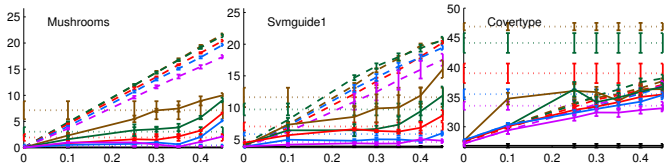


(e) Adversarial Noise in both classes

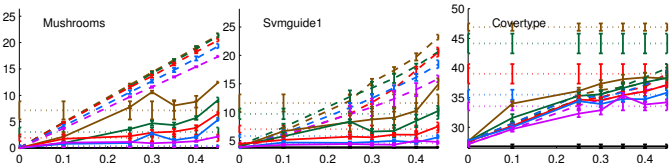
Figure 7: Plots show the effect of different fraction of expert labels for **LND** in terms of classification error of **SVM** trained with corrected label. In the plots the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in the training data-set.



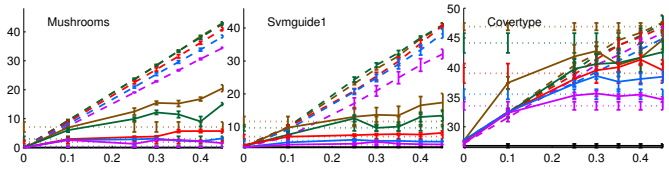
(a) Boundary Noise in majority class



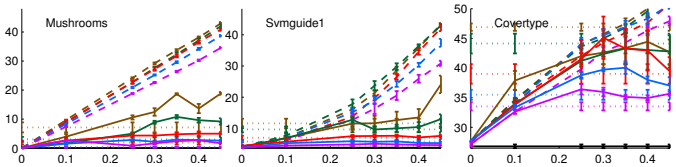
(b) Biased Annotator Noise in majority class



(c) Adversarial Noise in majority class

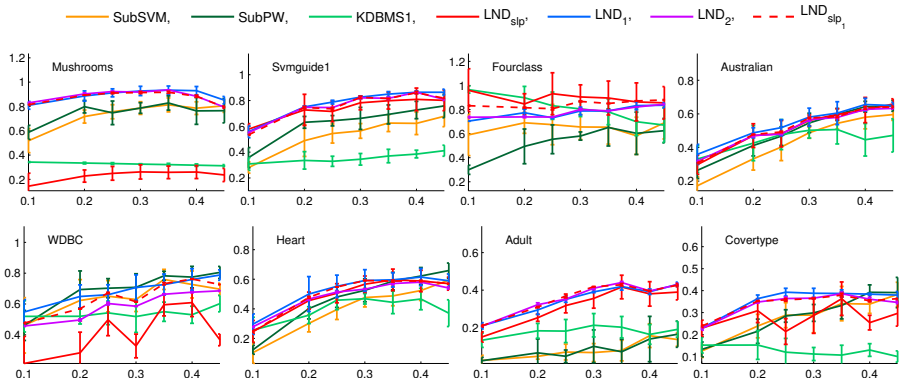


(d) Biased Annotator Noise in both classes

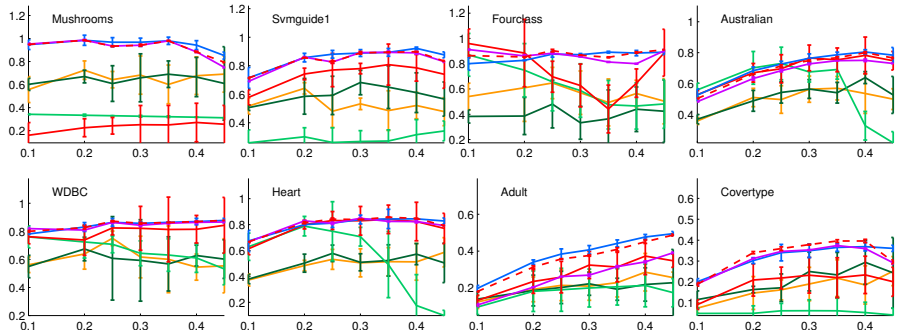


(e) Adversarial Noise in both classes

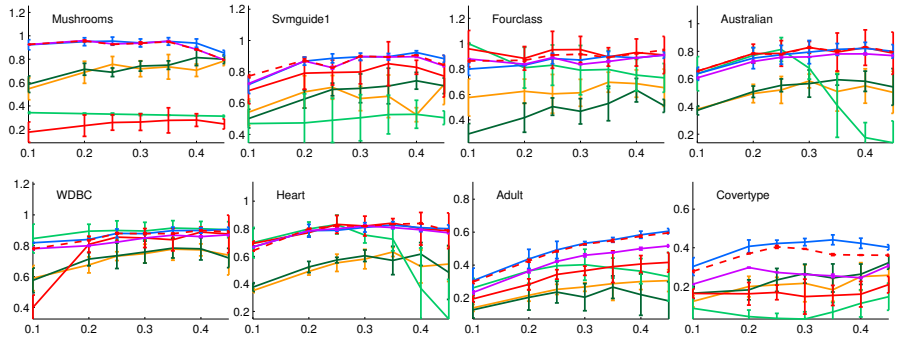
Figure 8: Plots show the effect of different fraction of expert labels for **LND** in terms of classification error of **PW** trained with corrected label. In the plots the y-axis shows the test error (in %) and the x-axis shows the fraction of noisy labels present in the training data-set.



(a) Boundary Noise

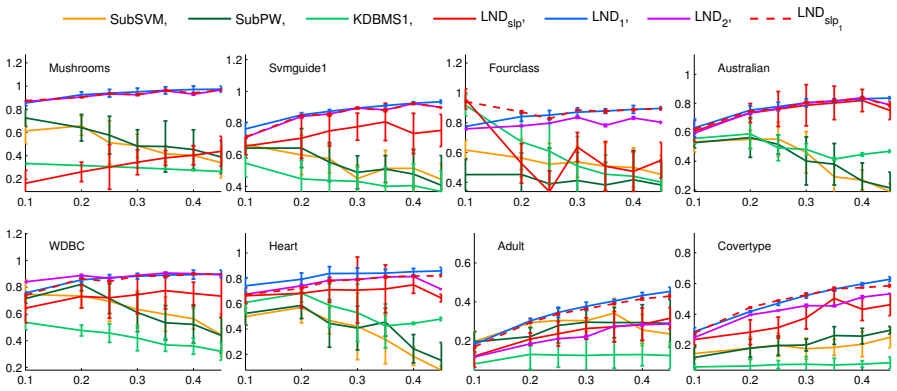


(b) Biased Annotator Noise

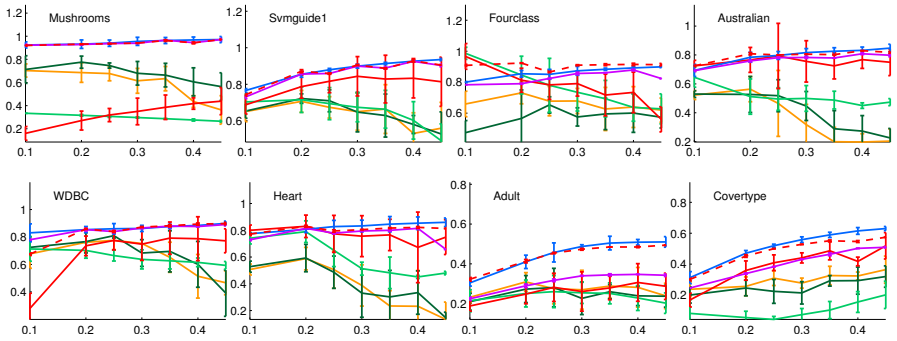


(c) Adversarial Noise

Figure 9: The label noise detecting ability of various methods in term of the F1-score. In the plot the y-axis shows the F1-score of noise detection and the x-axis shows the fraction of noisy labels present in the majority class of the training data-set.



(a) Biased Annotator Noise

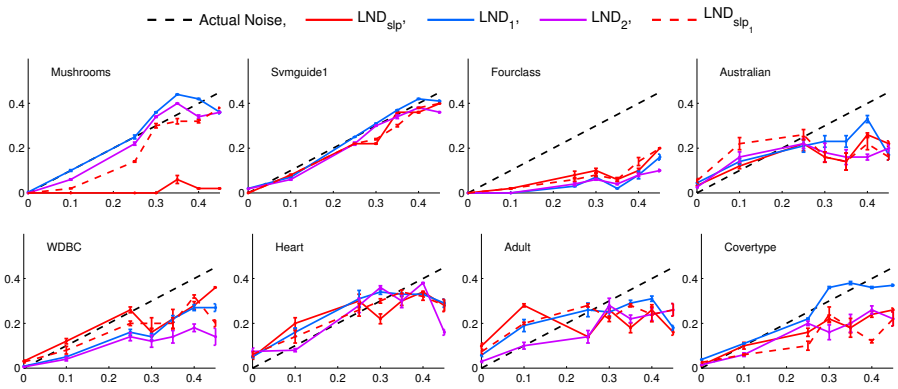


(b) Adversarial Noise

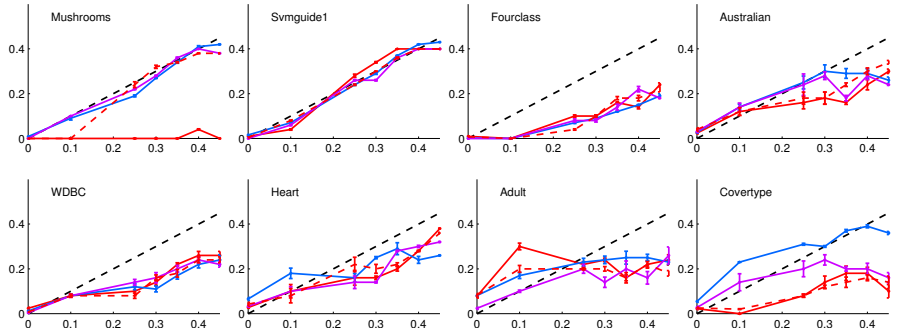
Figure 10: The label noise detecting ability of various methods in term of the F1-score. In the plot the y-axis shows the F1-score of noise detection and the x-axis shows the fraction of noisy labels present in both classes of the training data-set.

plot for the worst case scenario, i.e., the plots for no expert labels.

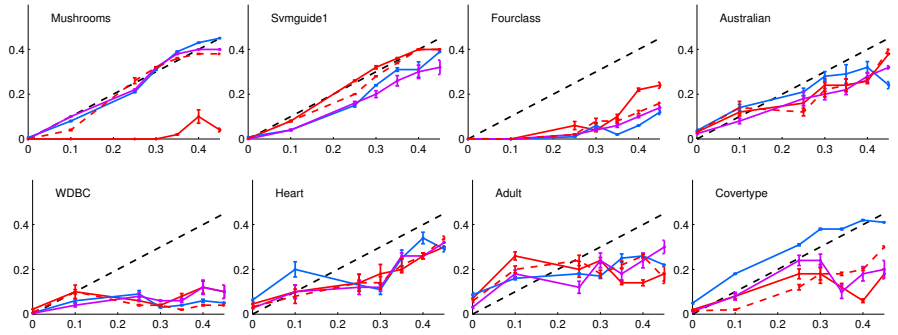
Noisy label detection ability: We compare in Figures 9, 10 the average F1 scores of detecting noisy labels achieved by different methods with proper parameter tuning through cross-validation. For this set of experiments we show the results only up-to rank-2 approximation of **LND** as for WDBC and Heart, the F1 score of rank-3 approximation is very close to that of the rank-2 approximation. In case of adversarial noise and biased annotator noise our proposed method is able to detect noisy labels with F1 scores higher than 0.8 except



(a) Boundary Noise

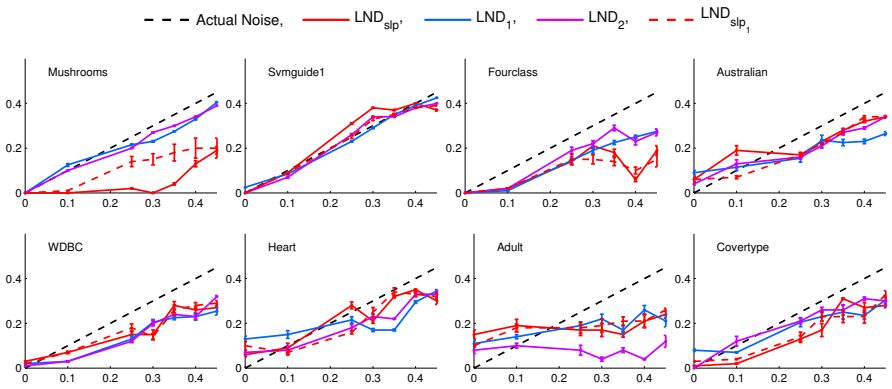


(b) Biased Annotator Noise

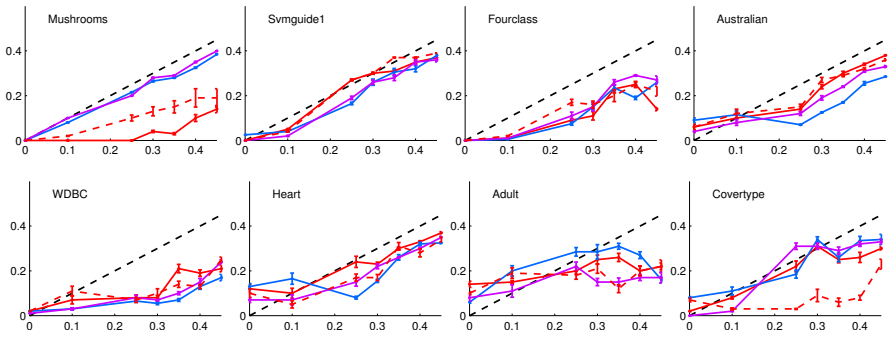


(c) Adversarial Noise

Figure 11: Accurate estimation of noise ratio. In each plot the y-axis shows the average estimation of noise ratio (ρ_+ or ρ_-) and the x-axis shows the fraction of noisy labels present in the majority class of the training data-set.



(a) Biased Annotator Noise

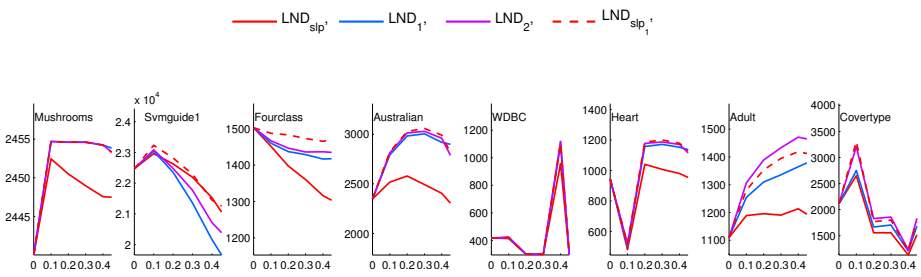


(b) Adversarial Noise

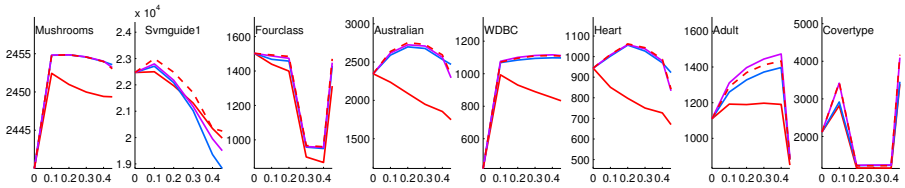
Figure 12: Accurate estimation of noise ratio. In each plot the y-axis shows the average estimation of noise ratio (ρ_+ or ρ_-) and the x-axis shows the fraction of noisy labels present in both classes of the training data-set.

for the Adult and the Covertype data-set. For the Mushrooms the F1 score is even closer to 1 for all kind of noise. Whereas the F1 scores for the other algorithms are below 0.6 in most of the cases. For small number of noisy labels, **KBDMS1** sometimes performs better than **LND** but when more noisy data points are present, $\mathbf{LND}_1$, $\mathbf{LND}_2$ and $\mathbf{LND}_{slp_1}$ outperform **KBDMS1** with a margin of more than 0.2. Performance of $\mathbf{LND}_{slp}$ is not consistent and varies a lot. This shows the superiority of Algorithm 4.

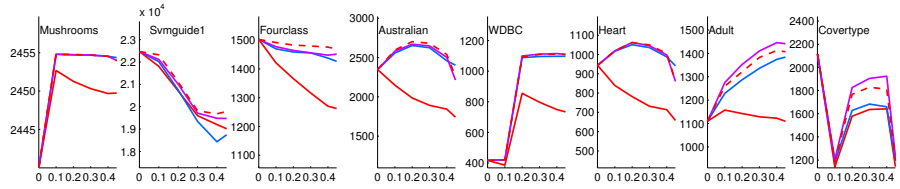
Accurate estimation of the noise ratio: We verify how close the estimated



(a) Boundary Noise



(b) Biased Annotator Noise

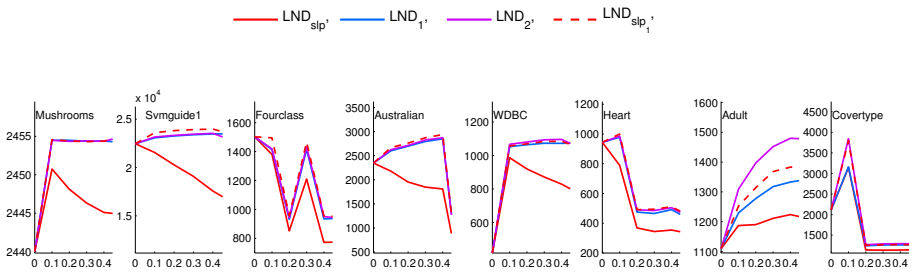


(c) Adversarial Noise

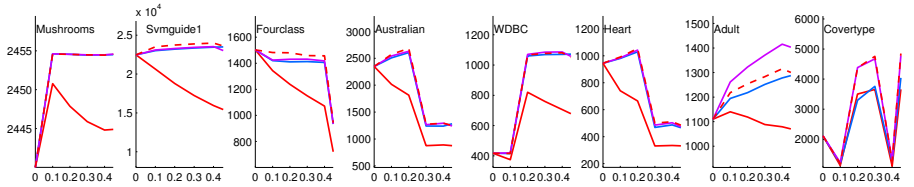
Figure 13: Plots show objective values achieved by various algorithms and approximations of **LND**, i.e., $\mathbf{LND}_{slp}$,$\mathbf{LND}_1$, $\mathbf{LND}_2$ and $\mathbf{LND}_{slp_1}$. In each plot the y-axis shows the objective values of **LND** (Equation (6)) and the x-axis shows the fraction of noisy labels present in the majority class of the training data-set.

noise ratios (ρ_+ and ρ_-) are to the actual percentage of noisy labels. Figures 11, 12 show that for proposed methods ($\mathbf{LND}_1$, $\mathbf{LND}_2$ and $\mathbf{LND}_{slp_1}$) the noise ratios are estimated accurately by cross-validation. For all data-sets except Adult and Fourclass and all types of noise models the estimated values are very close to the actual proportion of the noise. When the noise is larger (≥ 0.3) the estimated values are sometimes smaller than the true values.

Solution quality: We compare the objective values achieved by various al-



(a) Biased Annotator Noise

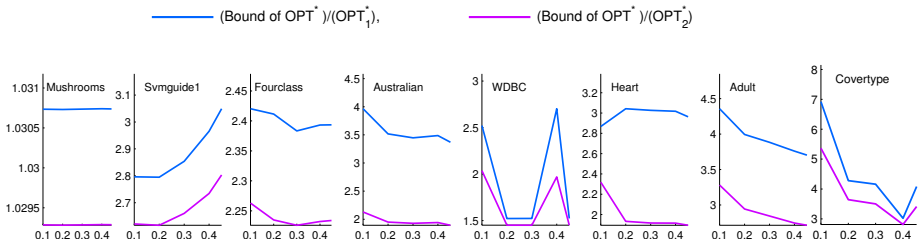


(b) Adversarial Noise

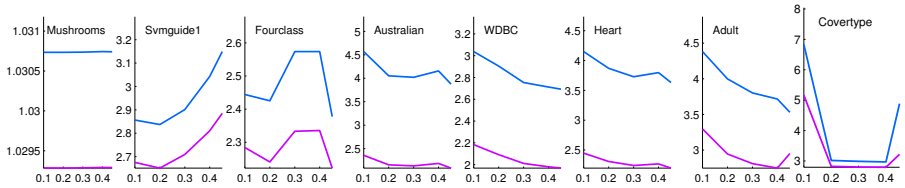
Figure 14: Plots show objective values achieved by various algorithms and approximations of **LND**, i.e., $\mathbf{LND}_{slp}$,$\mathbf{LND}_1$, $\mathbf{LND}_2$ and $\mathbf{LND}_{slp_1}$. In each plots the y-axis shows the objective values of **LND** (Equation (6)) and the x-axis shows the fraction of noisy labels present in both classes of the training data-set.

gorithms for solving (6). In these experiments, we use ρ_+ and ρ_- to be same as that used during the phase of insertion of noise. The experimental results (Figure 13,14 show that in almost all the cases the objective values achieved by $\mathbf{LND}_1$ and $\mathbf{LND}_2$ based on Algorithm 4 are much higher than that of $\mathbf{LND}_{slp}$. Whereas the objective values achieved by $\mathbf{LND}_{slp_1}$ is higher than that of $\mathbf{LND}_1$, and also for few data-sets (Svmguide1, Fourclass and Australian) objective values achieved by $\mathbf{LND}_1$ are even higher than $\mathbf{LND}_2$. Performance of $\mathbf{LND}_{slp}$ is not consistent as most of the time the objective values achieved by it are very far from the optimal one.

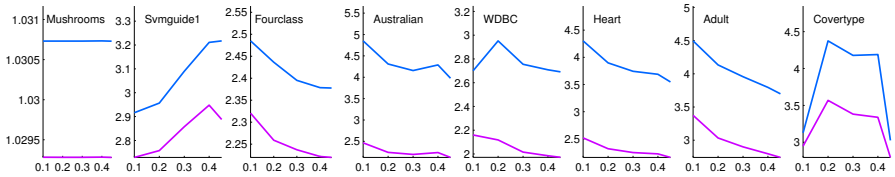
We plot (Figure 15,16) $\frac{\text{Bound of } OPT^*}{OPT^*_r}$, where the bound is calculated by (18). For the Mushrooms the bound is very tight with $\frac{\text{Bound of } OPT^*}{OPT^*_r} \leq 1.03$. In general, the approximation guarantees are very good and quite stable across all the data-sets.



(a) Boundary Noise



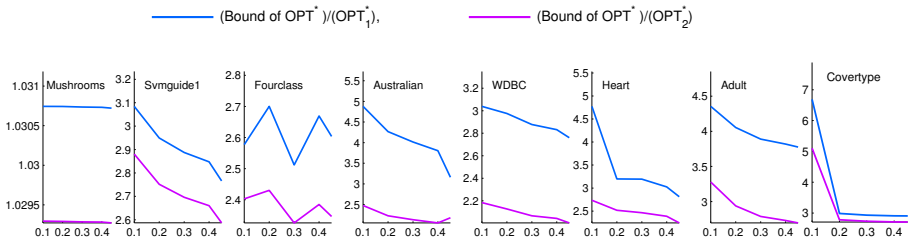
(b) Biased Annotator Noise



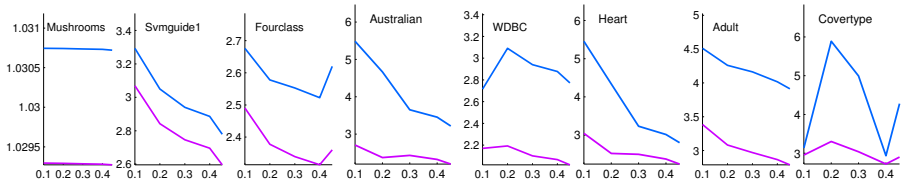
(c) Adversarial Noise

Figure 15: Plots show $\frac{\text{Bound of } OPT^*}{OPT^*_r}$ for $r = 1$ and 2. In each plot the y-axis shows $\frac{\text{Bound of } OPT^*}{OPT^*_r}$ and the x-axis shows the fraction of noisy labels present in the majority class of the training data-set.

Comparison of time complexity for noise label detection: Here we compare time complexities for various methods and algorithms. Each number in Table 2 reports average time required for each algorithm to detect noisy labels for a fixed set of parameter h, C, C_-, C_+ ρ_- and ρ_+. Time required for $\mathbf{LND}_1$ is less than that for **KBDMS1**, $\mathbf{LND}_{slp}$ and for smaller data sets it is comparable to **SubPW** and **SubSVM**. For moderately sized data-sets training time of $\mathbf{LND}_1$ is more than that of **SubPW** and **SubSVM**, where timing for **SubPW** and **SubSVM** depends on the number of local classifiers



(a) Biased Annotator Noise



(b) Adversarial Noise

Figure 16: Plots show $\frac{\text{Bound of } OPT^*}{OPT^*_r}$ for $r = 1$ and 2. In the plot the y-axis shows $\frac{\text{Bound of } OPT^*}{OPT^*_r}$ and the x-axis shows the fraction of noisy labels present in both classes of the training data-set.

learned. Here we learn **SubSVM** and **SubPW** with only 50 local classifiers. On the other hand, classification accuracy achieved by **SubSVM** and **SubPW** is much worse compared to that of $\mathbf{LND}_1$. As expected, the running time for $\mathbf{LND}_2$ is high but for larger data-sets it is better than the running time of **KBDMS1**. The time required for rank-3 approximation of **LND** for Heart and WDBC are 954.67 ± 106.2 and 591.23 ± 41.62 respectively.

In Figure 17 we study how the time complexities of noise detection algorithms vary with the increase of training data-sets. We do this using the Covertype data-set and we are not able to do this experiment beyond 3×10^4 for the proposed **LND** algorithms in our system.

7. Conclusion

In this paper we propose **LND** as a novel method for correcting labels of mislabeled data points. Although, the optimization problem underlying **LND** is NP-hard, we extend the Spannogram algorithm to obtain a solution with

Table 2: The time required for detecting noisy labels by various methods (in seconds)

Dataset	**SubSVM**	**SubPW**	**KBDMS1**	**LND**$_{slp}$	**LND**$_1$	**LND**$_2$	**LND**$_{slp_1}$
Heart	0.035	**0.007**	0.213	1.060	0.012	1.369	0.1473
150x13	±0.019	±0.002	±0.056	±0.245	±0.003	±0.571	±0.051
WDBC	0.034	**0.006**	0.291	2.249	0.013	2.826	0.0752
290x30	±0.012	±0.004	±0.060	±0.752	±0.0.002	±1.359	±0.015
Australian	0.047	**0.009**	0.358	2.510	0.013	2.081	0.0850
310x14	±0.015	±0.002	±0.155	±0.498	±0.002	±0.399	±0.015
Fourclass	0.054	**0.011**	0.621	3.8754	0.019	3.412	0.155
430x2	± 0.020	±0.003	±0.187	±0.834	±0.002	±0.282	±0.072
Adult	0.566	**0.155**	2.739	6.198	0.151	7.076	0.2538
1605x123	±0.394	±0.129	±0.681	±1.99	±0.016	±1.009	±0.543
Svmguide1	0.329	**0.297**	16.893	28.994	0.323	10.486	0.850
3089x4	±0.263	±0.133	±1.362	±5.405	±0.021	±1.380	±0.385
Mushroom	0.437	**0.350**	27.062	99.849	0.824	18.3610	2.8788
4062x112	±0.279	±0.085	±2.999	±17.813	±0.030	±1.885	±1.240
Covertype	0.502	**0.286**	41.386	156.517	1.394	35.266	3.702
5810x54	±0.277	± 0.053	± 3.845	±26.02	±0.0074	±2.111	±0.708

a provable approximation guarantee. Experimental results using a variety of data-sets and different noise models demonstrate that the proposed approach **LND** outperforms all existing methods for the detection of noisy labels by large margin. **SVM** trained with data pre-processed by **LND** also outperforms all other existing version of SVMs which are supposed to be robust to the label noise.

8. Acknowledgment

Sahely Bhadra is funded by the Indo-German Max Planck Center for Computer Science (IMPECS).

References

[1] J. Howe, Crowdsourcing: How the power of the crowd is driving the future of business, Random House, 2008.

[2] C. E. Brodley, M. A. Friedl, Identifying mislabeled training data, Journal of Artificial Intelligence Research 11 (1999) 131–167.

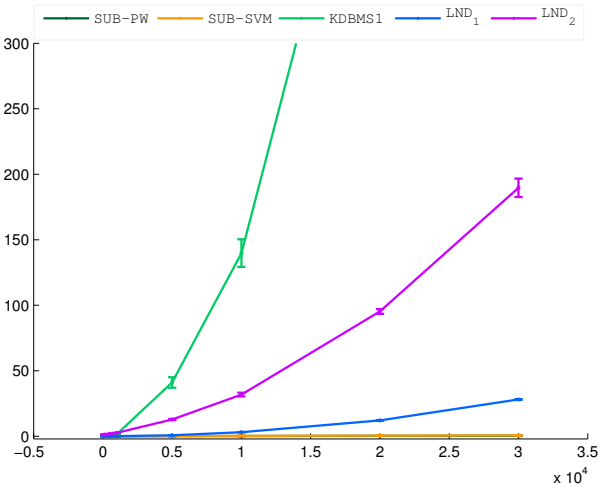


Figure 17: This plot compares the training time of various noise detection algorithms with various size of training data-sets. In the plot the y-axis shows the time required for detecting noisy labels for a fixed set of parameters (in *sec*) and the x-axis shows the number of training data points

[3] V. Guruswami, P. Raghavendra, Hardness of learning halfspaces with noise, in: FOCS, 2006, pp. 543–552.

[4] C. Caramanis, S. Mannor, Learning in the limit with adversarial disturbances, in: COLT, 2008, pp. 467–478.

[5] R. Collobert, F. Sinz, J. Weston, L. Bottou, Trading convexity for scalability, in: ICML, 2006, pp. 201–208.

[6] L. Xu, K. Crammer, D. Schuurmans, Robust support vector machine training via convex outlier ablation, in: AAAI, 2006, pp. 536–542.

[7] G. Stempfel, L. Ralaivola, Learning svms from sloppily labeled data, in: ICANN (1), 2009, pp. 884–893.

[8] B. Biggio, B. Nelson, P. Laskov, Support vector machines under adversarial label noise, in: ACML, Vol. 20, 2011, pp. 97–112.

[9] N. Natarajan, I. Dhillon, P. Ravikumar, A. Tewari, Learning with noisy labels, in: NIPS, 2013, pp. 1196–1204.

[10] H. Valizadegan, P. N. Tan, Kernel based detection of mislabeled training examples., in: SDM, 2007.

[11] J. Cao, S. Kwong, R. Wang, A noise-detection based adaboost algorithm for mislabeled data, Pattern Recognition 45 (12) (2012) 4451–4465.

[12] S. Laxman, S. Mittal, R. Venkatesan, Error correction in learning using svms, CoRR abs/1301.2012.

[13] D. S. Papailiopoulos, A. G. Dimakis, S. Korokythakis, Sparse pca through low-rank approximations, in: ICML, Vol. 3, 2013, pp. 747–755.

[14] D. S. Papailiopoulos, I. Mitliagkas, A. Dimakis, C. Caramanis, Finding dense subgraphs through low-rank approximations, preprint.
URL `https://webspace.utexas.edu/dp26726/papers/DkS_long.pdf`

[15] X. Zhu, X. Wu, Q. Chen, Eliminating class noise in large datasets, in: ICML, 2003, pp. 920–927.

[16] R. Duda, P. Hart, D. Stork, Pattern Classification, Wiley, 2001.

[17] D. Wied, R. Weißbach, Consistency of the kernel density estimator: a survey, Statistical Papers 53 (1) (2012) 1–21.

[18] B. Moghaddam, Y. Weiss, S. Avidan, Spectral bounds for sparse pca: Exact and greedy algorithms, in: Advances in neural information processing systems, 2005, pp. 915–922.

[19] C. Chang, C. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 27:1–27:27, dataset available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets`.