

Spectral Label Refinement for Noisy and Missing Text Labels

Yangqiu Song^a Chenguang Wang^b Ming Zhang^b Hailong Sun^c Qiang Yang^d

^aUniversity of Illinois at Urbana-Champaign ^bPeking University

^cBeihang University ^dHong Kong University of Science and Technology

^ayqsong@illinois.edu ^b{wangchenguang,mzhang_cs}@pku.edu.cn ^csunhl@act.buaa.edu.cn ^dqyang@cse.ust.hk

Abstract

With the recent growth of online content on the Web, there have been more user generated data with noisy and missing labels, e.g., social tags and voted labels from Amazon’s Mechanical Turks. Most of machine learning methods, which require accurate label sets, could not be trusted when the label sets were yet unreliable. In this paper, we provide a text label refinement algorithm to adjust the labels for such noisy and missing labeled datasets. We assume that the labeled sets can be refined based on the labels with certain confidence, and the similarity between data being consistent with the labels. We propose a label smoothness ratio criterion to measure the smoothness of the labels and the consistency between labels and data. We demonstrate the effectiveness of the label refining algorithm on eight labeled document datasets, and validate that the results are useful for generating better labels.

Introduction

With the recent growth of the online content generation, there are lots of datasets with noisy and missing labels. Supervised machine learning methods, such as classification and ranking, have demonstrated their effectiveness in broad applications, such as recommendation systems, natural language processing tasks. On one hand, the more labeled and accurate label sets are input to a supervised learning method, the more improvement on the performance one can gain. On the other hand, noisy and missing labels could hurt the performance in a considerable way with different learning algorithms, e.g., naive Bayes being better than support vector machines with sequential minimal optimization (SMO) trained on noisy labels (Nettleton, Orriols-Puig, and Fornells 2010). However, in real world, the situation can be even worse. The labeled data on the Web can be extremely noisy and missing.

For example, online crowdsourcing systems such as Amazon’s Mechanical Turk¹ and Rent-A-Coder² can facilitate the labeling tasks, by matching “labelers” with well defined “tasks.” However, since the labelers may lack expertise, dedication, and interest, the resulting labels are often noisy and will affect the decisions of learners (Raykar et

al. 2010). Even with certain processing of the labels annotated by the non-expert labelers, such as voting, the resulting labels could be still noisy (Sheng, Provost, and Ipeirotis 2008). Moreover, in social networks, such as Facebook and Twitter, users are often allowed to provide certain tags or profile information to gain attention from the others sharing the similar interests. However, not all of the users want to publicly annotate their private profile information. In addition, the provided labels could be very noisy (Law, Settles, and Mitchell 2010), since different users have different habits or preferences. For example, for the labels “movie” and “film,” they are same, but can appear in two users’ tags. Another example is that a user may be an expert on artificial intelligence and she tags herself with the term, but she only publishes movie related content. In this case, the tag does not perfectly characterize the contents that are published. Thus, noisy and missing labels are common in social networks. Furthermore, traditional natural language processing (NLP) tasks can also benefit from noisy data labeled by non-experts, as if there are some mechanisms to reduce the label noise (Pal, Mann, and Minerich 2007; Snow et al. 2008). However, in some of more difficult tasks, such as event extraction, the mutual agreement of human labels is only around 40 – 50% (Ji and Grishman 2008). In such cases, non-expert annotations could be much worse. Therefore, all the above examples indicate that more effective algorithms to deal with the noisy and missing label problem should be developed.

In this paper, we deal with the noisy and missing label problem with a label refinement mechanism. Instead of proposing a supervised learning algorithm that can handle the noise, we propose an algorithm that can modify the labels themselves. Then the refined labels can be used for other machine learning and data mining tasks. With the assumption that *the data samples are static and i.i.d., and the labels of data are consistent with their nearest neighborhoods*, we propose a label smoothness ratio criterion to refine the noisy and missing labels. Our approach considers both the content of data (by constraining the refined labels to be smooth on content graph) and the initial labels (by constraining the refined label being smooth on the graph constructed by the initial labels). We relax the estimated labels to the real values and use spectral analysis to solve the problem. The final solution is given by a generalized eigenvalue

decomposition problem. We also provide a rotation algorithm to align the estimated eigenvectors with the provided labels. Experiments conducted on eight real world datasets have shown its power in following three aspects.

- Our approach is able to refine the noisy labels. We tested on the datasets by randomly generating labels.
- Our approach is able to refine the missing labels by completing the label sets. This is similar to semi-supervised learning (Chapelle, Schölkopf, and Zien 2006).
- Our approach is also able to refine the clustering results of other clustering algorithms. After pre-clustering using the state-of-the-art clustering algorithms, our approach can significantly improve the clustering results.

Related Work

In this section, we review some related work on multiple noisy labels voting and machine learning algorithms for noisy labels.

The first research direction mainly focuses on using the labels from multiple noisy labelers to refine the labels. Voting is widely used for the dataset when multiple noisy labelers are available (Sheng, Provost, and Ipeirotis 2008). To refine labels based on multiple noisy labelers, Snow et al. (2008) and Raykar et al. (2010) used a Bayesian model to show that by modeling multiple labelers one can obtain labels as accurate as some experts. Whitehill et al. (2009) proposed to use a Bayesian algorithm to handle the labeler's expertise and task difficulty in voting. Zhou et al. (2012) proposed a maximum entropy framework to solve the same labeler's expertise and task difficulty problem. However, all of the above approaches assume that each data should be labeled with multiple labelers. In contrast, we do not need to acquire multiple labels. Our approach could be further applied to crowdsourcing systems asking for only one label per data sample, which can save a lot of labor and money in practice.

The other research direction is learning a better classifier from the noisy labels. Zhu, Wu, and Chen (2003) proposed a rule based algorithm to identify the noise in the labels by training different subsets of the labeled data. While rule based system has high accuracy to model the detected noise pattern, it may not be able to generalize to other noise patterns. Ramakrishnan et al. (2005) and Yang et al. (2012) tried to use a classifier that allows labels to have noise, and provide either a probabilistic inference or a stochastic programming to iteratively learn a better model. Li et al. (2013) proposed an interesting framework that can incorporate the label distance into the multi-label learning framework. Although the problem is multi-label learning, the algorithm should reduce it to a set of binary classification problems. Moreover, Natarajan et al. (2013) provided a more theoretical analysis about the cost function of the noisy label problem, and provided a surrogate loss to learn the problem. Most the above research essentially works for binary classification problem, and needs extra efforts to extend to multi-class classification. Different from them, we use an algorithm based on clustering, i.e., spectral clustering (Ng, Jordan, and Weiss 2001), to refine labels automatically. In this case, we are not restricted to binary classification problem,

and can be easily applied to multi-class classification. Moreover, by checking the consistency between the data content and the initial noisy labels, we have a closed-form solution based on the generalized eigenvalue decomposition, which is much easier to implement in practice.

Data-Label Smoothness Ratio-based Label Refinement

In this section, we introduce the label refinement approach using the smoothness ratio criteria defined on data and label similarities. We denote the input dataset as $\mathcal{S} = \{\mathcal{X}, \mathcal{Y}\}$. The feature set is denoted as $\mathcal{X}$, where $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. Each data sample $\mathbf{x} \in \mathbb{R}^M$ is an M -dimensional vector. Their corresponding labels are $\mathcal{Y} = \{y_1, y_2, \dots, y_N\}$, which are noisy or partially missing.

Label Smoothness based on Data Similarity

Inspired by normalized cut used for clustering (Shi and Malik 2000), the original binary clustering algorithm is done by partitioning the nodes $\mathcal{V}$ of graph $\mathcal{G}$ into two disjoint parts $\mathcal{A}$ and $\mathcal{B}$, where $\mathcal{A} \cap \mathcal{B} = \emptyset$ and $\mathcal{A} \cup \mathcal{B} = \mathcal{V}$. We build a k -nearest-neighbor graph based on the data similarity. We denote $\mathbf{W}$ as the adjacency matrix of the graph, where $\mathbf{W}_{ij}$ is the weight on the edge between nodes i and j . We use the self tuning local scaling approach to compute the weights $\mathbf{W}_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_i\sigma_j}\right)$, where σ_i is the distance from $\mathbf{x}_i$ to its $\lfloor k/2 \rfloor^{th}$ nearest neighbors (Zelnik-manor and Perona 2004). Let $cut(\mathcal{A}, \mathcal{B}) = \sum_{i \in \mathcal{A}, j \in \mathcal{B}} \mathbf{W}_{ij}$ denote the sum of the weights between $\mathcal{A}$ and $\mathcal{B}$, and $assoc(\mathcal{A}, \mathcal{V}) = \sum_{i \in \mathcal{A}, j \in \mathcal{V}} \mathbf{W}_{ij} = \sum_{i \in \mathcal{A}} d_i$, is the connection between the nodes in $\mathcal{A}$ to all the node in $\mathcal{V}$, where $d_i = \sum_{j=1}^N \mathbf{W}_{ij}$ is the degree of vertex $\mathbf{x}_i$. The normalized cut criterion is represented by:

$$\mathcal{J}_{NCut}(\mathcal{A}, \mathcal{B}) = \frac{cut(\mathcal{A}, \mathcal{B})}{assoc(\mathcal{A}, \mathcal{V})} + \frac{cut(\mathcal{B}, \mathcal{A})}{assoc(\mathcal{B}, \mathcal{V})} \quad (1)$$

The partition is desired to find the subsets $\mathcal{A}$ and $\mathcal{B}$ such that the normalized cut criterion $\mathcal{J}_{NCut}(\mathcal{A}, \mathcal{B})$ is minimized. By defining the normalized graph Laplacian (Chung 1997):

$$\bar{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}}, \quad (2)$$

where the diagonal matrix $\mathbf{D}$ satisfies $\mathbf{D}_{ii} = d_i$, it has been shown that the solution is given by optimizing the following criterion (Shi and Malik 2000):

$$\mathbf{f}_{\bar{\mathbf{L}}}^* = \underset{s.t. \mathbf{f}^T \mathbf{f}_0 = 0}{\operatorname{argmin}} \frac{\mathbf{f}^T \bar{\mathbf{L}} \mathbf{f}}{\mathbf{f}^T \mathbf{D} \mathbf{f}} \quad (3)$$

where $\mathbf{f} = (f_1, f_2, \dots, f_N)^T$ is denoted as the relaxed labels. This can be solved by finding the second smallest eigenvector of the generalized system $\bar{\mathbf{L}} \mathbf{f} = \lambda \mathbf{D} \mathbf{f}$ and $\mathbf{f}_0 = \mathbf{1}$ is the eigenvector corresponding to the smallest eigenvalue $\lambda_0 = 0$. This is equal to first optimize

$$\mathbf{f}_{\bar{\mathbf{L}}}^* = \underset{s.t. \mathbf{f}^T \mathbf{f}_0 = 0}{\operatorname{argmin}} \frac{\mathbf{f}^T \bar{\mathbf{L}} \mathbf{f}}{\mathbf{f}^T \mathbf{f}}. \quad (4)$$

Then the two solutions (3) and (4) have the connection as $\mathbf{f}_L^* = \mathbf{D}_L^{\frac{1}{2}} \mathbf{f}_L^*$. Intuitively, this smoothness criterion constrains the learning function from being changed much from the nearby points. For the similar nodes i and j on graph, it imposes a large weight $\mathbf{W}_{ij}$ related to the difference between f_i and f_j .

Label Smoothness based on Noisy Label Similarity

The graph Laplacian view of normalized cut gives us a good inspiration that the smoothness is an important criterion to find good partition of a graph. For the data with initial noisy labels, we can also define a graph based on the labels.

Definition 1 The adjacency matrix is defined with respect to the similarity of the initial labels:

$$\mathbf{A}_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ have the same label} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

We have:

$$\bar{\mathbf{L}}_{\mathbf{A}} = \mathbf{I} - \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}}, \quad (6)$$

where $\bar{\mathbf{L}}_{\mathbf{A}}$ is the normalized graph Laplacian associated with the label similarity based adjacency matrix, $\mathbf{D}_{\mathbf{A}ii} = d_{\mathbf{A}i}$, and $d_{\mathbf{A}i} = \sum_{j=1}^N \mathbf{A}_{ij}$ is the number of labels in one category i .

Similar to the content based graph, label similarity based graph also makes the function smooth if the initial labels are similar. Also, if one category i has few item, which means $d_{\mathbf{A}i}$ is small, then the criterion puts a large weight $1/\sqrt{d_{\mathbf{A}i}}$ to the minimization function. Therefore, the label effect is normalized, and category with smaller size has bigger chance to be identified.

Label Smoothness based on Data-Label Joint Similarity

Given the assumption that the labels are noisy or missing, we can not fully trust the initial labels. Therefore, we propose to use the following *data-label smoothness ratio (DLSR)* criterion to identify the true labels. In this criterion, both of the content information and the label information are jointly used to obtain better labels.

Note the following fact:

$$\begin{aligned} \operatorname{argmin}_{\mathbf{f}} \frac{\mathbf{f}^T \bar{\mathbf{L}}_{\mathbf{A}} \mathbf{f}}{\mathbf{f}^T \mathbf{f}} &= \operatorname{argmin}_{\mathbf{f}} \frac{\mathbf{f}^T (\mathbf{I} - \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}}) \mathbf{f}}{\mathbf{f}^T \mathbf{f}} \\ &= \operatorname{argmax}_{\mathbf{f}} \frac{\mathbf{f}^T \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}} \mathbf{f}}{\mathbf{f}^T \mathbf{f}} \\ &= \operatorname{argmax}_{\mathbf{f}} \frac{\mathbf{f}^T \bar{\mathbf{A}} \mathbf{f}}{\mathbf{f}^T \mathbf{f}} \end{aligned} \quad (7)$$

where $\bar{\mathbf{A}} = \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}}$. Given both data content and initial labels, we should find a set of soft labels that minimizes the term in (4) and maximizes the term in (7) simultaneously. By combining both of them, we propose to use the following criterion:

$$\mathbf{f}_{Sim}^* = \operatorname{argmin}_{\mathbf{f}} \frac{\mathbf{f}^T \bar{\mathbf{L}} \mathbf{f}}{\mathbf{f}^T \bar{\mathbf{A}} \mathbf{f}} \quad (8)$$

This leads to a generalized eigenvalue decomposition problem:

$$\bar{\mathbf{L}} \mathbf{f}_{Sim}^* = \lambda_{Sim}^* \bar{\mathbf{A}} \mathbf{f}_{Sim}^* \quad (9)$$

where λ_{Sim}^* is the smallest generalized eigenvalue and $\mathbf{f}_{Sim}^*$ is the corresponding eigenvector.

Besides label similarity, we can also incorporate the dissimilarity between different categories. Following (Goldberg, Zhu, and Wright 2007), dissimilarity can be represented in a graph Laplacian view based on the following definitions.

Definition 2 If two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar, the objective function is $(f_i - f_j)^2$; If two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ are dissimilar, the objective is $(f_i + f_j)^2$. By introducing a tuning parameter, the objective function can be written as $(f_i - s f_j)^2$. For similarity, $s = 1$, for dissimilarity, $s = -1$.

Definition 3 A mixed graph is defined based on the similarity and dissimilarity of labels. It can be represented by the matrix $\mathbf{B}$ satisfying

$$\mathbf{B}_{ij} = \begin{cases} a & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \text{ have the same label} \\ -b & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \text{ have the different labels} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

where a and b are the coefficients that control the balance of similarity and dissimilarity. The degree of each node is defined as:

$$d_{\mathbf{B}i} = \sum_{j=1}^N |\mathbf{B}_{ij}|. \quad (11)$$

Moreover, the normalized graph Laplacian is

$$\bar{\mathbf{L}}_{\mathbf{B}} = \mathbf{I} - \mathbf{D}_{\mathbf{B}}^{-\frac{1}{2}} \mathbf{B} \mathbf{D}_{\mathbf{B}}^{-\frac{1}{2}}, \quad (12)$$

where $\mathbf{D}_{\mathbf{B}ii} = d_{\mathbf{B}i}$.

It is not difficult to verify that the above normalized graph Laplacian is positive semi-definite where

$$\mathbf{f}^T \bar{\mathbf{L}}_{\mathbf{B}} \mathbf{f} = \sum_{ij} |\mathbf{B}_{ij}| \left(\frac{f_i}{\sqrt{d_{\mathbf{B}i}}} - \frac{\mathbf{B}_{ij}}{|\mathbf{B}_{ij}|} \frac{f_j}{\sqrt{d_{\mathbf{B}j}}} \right)^2 \geq 0. \quad (13)$$

Following the derivation above, we have the objective function as

$$\mathbf{f}_{S-Diss}^* = \operatorname{argmin}_{\mathbf{f}} \frac{\mathbf{f}^T \bar{\mathbf{L}} \mathbf{f}}{\mathbf{f}^T \bar{\mathbf{B}} \mathbf{f}} \quad (14)$$

where $\bar{\mathbf{B}} = \mathbf{D}_{\mathbf{B}}^{-\frac{1}{2}} \mathbf{B} \mathbf{D}_{\mathbf{B}}^{-\frac{1}{2}}$. This leads to a generalized eigenvalue decomposition problem:

$$\bar{\mathbf{L}} \mathbf{f}_{S-Diss}^* = \lambda_{S-Diss}^* \bar{\mathbf{B}} \mathbf{f}_{S-Diss}^* \quad (15)$$

where λ_{S-Diss}^* is the smallest generalized eigenvalue and $\mathbf{f}_{S-Diss}^*$ is the corresponding eigenvector.

Label Alignment

The above analysis is all about binary problem. For the N -way case, there have been many approaches. For example, we can use the recursive 2-way clustering algorithm to partition the data $N - 1$ times (Shi and Malik 2000), or use

Algorithm 1 *DLSR*-based Label Refinement Algorithm

1. Find partition matrix:
 - Compute the normalized graph Laplacian $\bar{\mathbf{L}}$ in Eq. (2) based on document content;
 - Compute the normalized matrix $\bar{\mathbf{A}} = \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_{\mathbf{A}}^{-\frac{1}{2}}$ or $\bar{\mathbf{B}} = \mathbf{D}_{\mathbf{B}}^{-\frac{1}{2}} \mathbf{B} \mathbf{D}_{\mathbf{B}}^{-\frac{1}{2}}$ based on labels;
 - Solve the generalized eigenvalue decomposition problem $\bar{\mathbf{L}} \mathbf{f}_{Sim}^* = \lambda_{Sim}^* \bar{\mathbf{A}}$ or $\bar{\mathbf{L}} \mathbf{f}_{S-DisS}^* = \lambda_{S-DisS}^* \bar{\mathbf{B}}$;
 - Obtain $\mathbf{F}_{Sim}^*$ or $\mathbf{F}_{S-DisS}^*$;
 2. Find discretized solution:
 - Compute $\mathbf{H}^* = \text{diag}(\mathbf{F}^* \mathbf{F}^{*T})^{-1/2} \mathbf{F}^*$;
 - Minimize $\|\mathbf{H} - \mathbf{H}^* \mathbf{R}\|$ in Eq. (18);
- Return $\hat{\mathbf{H}}^*$.
-

clustering algorithm, such as Kmeans, to cluster the embedded points in the eigenvector space (Ng, Jordan, and Weiss 2001). Here, we use an optimization approach to align the refined labels and the initial labels (Yu and Shi 2003).

We first define an indicator/partition matrix $\mathbf{H} \in \mathbb{R}^{N \times K}$ whose elements are $\mathbf{H}_{ij} = 1$ if document d_i belongs to the j^{th} class ($1 \leq j \leq K$), K is the number of clusters, and $\mathbf{H}_{ij} = 0$ otherwise. For each row of $\mathbf{H}$, there is one and only one element equals to 1. Then the scaled partition matrix is define as $\mathbf{F} = \mathbf{H}(\mathbf{H}^T \mathbf{H})^{-1/2}$, such that $\mathbf{F}^T \mathbf{F} = \mathbf{I}$ where $\mathbf{I}$ is an identity matrix. Given a scaled partition matrix $\mathbf{F}$, the original partition matrix is given by $\mathbf{H} = \text{diag}(\mathbf{F} \mathbf{F}^T)^{-1/2} \mathbf{F}$. We employ the generalized eigenvalue decomposition method to find the scaled partition matrix, which is $\mathbf{F} = (\mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \dots, \mathbf{f}^{(K)})$ where $\mathbf{f}^{(i)} = (f_1^{(i)}, f_2^{(i)}, \dots, f_N^{(i)})^T$. Then we use another optimization method to estimate the discretized partition matrix $\mathbf{H}$.

Specifically, we relax the scaled partition matrix $\mathbf{F}$ to the continuous soft labels, and optimize the following objective:

$$\mathbf{F}_{Sim}^* = \underset{s.t. \mathbf{F}^T \mathbf{F} = \mathbf{I}}{\text{argmin}} \frac{\det(\mathbf{F}^T \bar{\mathbf{L}} \mathbf{F})}{\det(\mathbf{F}^T \bar{\mathbf{A}} \mathbf{F})} \quad (16)$$

for the similarity based algorithm, where $\det(\cdot)$ is the determinant of a matrix. The solution is given by the generalized eigenvalue decomposition problem:

$$\bar{\mathbf{L}} \mathbf{f}^{(i)*} = \lambda_i^* \bar{\mathbf{A}} \mathbf{f}^{(i)*} \quad (17)$$

where $\mathbf{f}^{(i)*}$'s are the eigenvectors corresponding to the first K smallest eigenvectors of λ_i^* 's. For the case which involves both the similarity and dissimilarity of labels, we replace the matrix $\bar{\mathbf{A}}$ with $\bar{\mathbf{B}}$.

Thus, we obtain the approximated optimal scaled partition matrix $\mathbf{F}^* = (\mathbf{f}^{(1)*}, \mathbf{f}^{(2)*}, \dots, \mathbf{f}^{(K)*})$. It is known that the optimal solution is not unique. Instead, it is in the space spanned by $\{\mathbf{f}^{(1)*}, \mathbf{f}^{(2)*}, \dots, \mathbf{f}^{(K)*}\}$. This means that, for any orthogonal matrix $\mathbf{R}$ (such that $\mathbf{R} \in \mathbb{R}^{K \times K}$ and $\mathbf{R}^T \mathbf{R} = \mathbf{I}$), $\mathbf{F}^* \mathbf{R}$ is also an optimal solution. Replace $\mathbf{F}^*$ with $\mathbf{F}^* \mathbf{R}$ in $\mathbf{H}^* = \text{diag}(\mathbf{F}^* \mathbf{F}^{*T})^{-1/2} \mathbf{F}^*$, we see that $\mathbf{H}^* \mathbf{R}$ is near to true labels. Therefore, we use the following

Table 1: A summary of datasets. The balance is defined as the ratio of the number of documents in smallest class to the one of the largest class.

Data	#Docs	#Words	#Class	#Avg Doc	Balance
tr11	414	6,429	9	46	0.046
tr12	313	5,799	8	39	0.097
tr23	204	5,832	6	34	0.066
tr31	927	10,127	8	128	0.006
tr41	878	7,454	10	88	0.037
tr45	690	8,261	10	69	0.088
ohscal	11,162	11,465	10	1,116	0.437
NG20	19,949	43,586	20	997	0.991

optimization problem to obtain $\mathbf{H}$ (Yu and Shi 2003):

$$\hat{\mathbf{H}} = \underset{\mathbf{H} \in \mathbb{R}^{N \times K}, \mathbf{R} \in \mathbb{R}^{K \times K}}{\text{argmin}} \|\mathbf{H} - \mathbf{H}^* \mathbf{R}\|$$

$$s.t. \quad \mathbf{H}_{ij} \in \{0, 1\}, \mathbf{H} \mathbf{1}_K = \mathbf{1}_N, \mathbf{R}^T \mathbf{R} = \mathbf{I}. \quad (18)$$

where $\mathbf{H}^* = \text{diag}(\mathbf{F}^* \mathbf{F}^{*T})^{-1/2} \mathbf{F}^*$. $\mathbf{H}$ is initialized with the noisy labels. $\mathbf{1}_N$ and $\mathbf{1}_K$ are vectors with all one elements.

The above label refinement algorithm is shown in Algorithm 1.

Experiments

In this section, we compare our *DLSR*-based label refinement algorithm (short as *DLSR*) (Algorithm 1) with the state-of-the-art clustering algorithms. First, we introduce the datasets we used.

Datasets and Evaluation

To evaluate our algorithm, we use eight text classification datasets that containing the ground truth labels. Specifically, we use the datasets presented in (Zhong and Ghosh 2005), which are the 20-newsgroups data and the sets from the CLUTO toolkit (Karypis 2002). Eight subsets are selected to test our algorithm, which are summarized in Table 1.

The NG20 dataset represents the 20-newsgroups data. It collects 20,000 messages of 20 different newsgroups. The data was preprocessed by the Bow toolkit (McCallum 1996). The data was chopped off the headers, removed stopwords and the words occur in less than three documents (Zhong and Ghosh 2005). Then the document is represented by a feature with 43,586 dimensional sparse vector. Several empty documents were also removed (Zhong and Ghosh 2005). All the datasets used in CLUTO were first preprocessed (Zhao and Karypis 2001) and then processed by removing the words appear in two or fewer documents (Zhong and Ghosh 2005). The ohscal dataset is from OHSUMED collection (Hersh et al. 1994). Datasets tr11, tr12, tr23, tr31, tr41 and tr45 are from TREC collections³. All the data are computed using normalized *TF-IDF* feature. The neighborhood number to construct the content based neighborhood graphs for all the graph based algorithms is empirically set to 10.

³<http://trec.nist.gov>

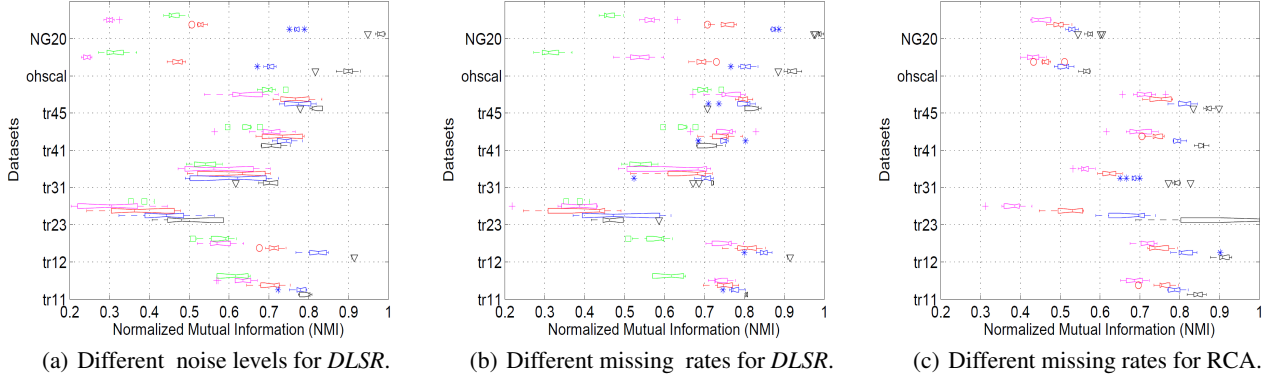


Figure 1: The whiskers are lines extending from each end of the box to show the extent of the rest of the data. Outliers are data with values beyond the ends of the whiskers, which are displayed by several symbols. The noise/missing rates are 0% (color black, symbol triangle), 20% (color blue, symbol star), 40% (color red, symbol circle), 60% (color magenta, symbol plus sign) respectively. In (a) and (b), we also provide the results of traditional normalized cut algorithm, which is shown as green square.

For comparison of different results, we select Normalized Mutual Information (NMI) as the performance measure. The NMI score is 1 if the refined labels perfectly match the ground truth labels and it being 0 means random labeled. Thus, the larger score, the better the label refinement result is. All the NMI scores reported are based on 50 runs.

Noisy Label Refinement

We first test our *DLSR* algorithm with different noise rates for labels. We set the initial labels of data as the ground truth labels. Then, we add some noises on these labels. For example, the noise rate 40% represents that we randomly select 40% of the true labels and randomly permute these labels. Here, we set the noise rates as 0%, 20%, 40% and 60%. We set $a = 1$ and $b = 0.001$ (defined in Definition 3) for this experiment. The results for the eight datasets are shown in Fig. 1(a).

It is shown that the label noises affect the NMI results. More noises make the results worse. The results without any noise (0% noise) are the best. With 20% and 40% noise, our algorithm can refine the initial labels and perform better than traditional normalized cut (*NCut*) (Shi and Malik 2000) algorithm. When there are more noisy labels in the data (i.e., 60%), the accuracy rates may be lower than the *NCut* algorithm for some datasets. We conclude that *DLSR* does not completely trust the labels and can refine some of them, while very large amount of incorrect labels can still mislead the label refinement result.

Missing Label Refinement

We then test our algorithm with partially missing labeled data, by randomly changing different portion of labeled data to unlabeled ones. The missing label rates are set to be 0%, 20%, 40% and 60%. We also set $a = 1$ and $b = 0.001$ for this experiment. The missing label result is shown in Fig. 1(b). Overall, more missing labels will lead to the worse results. Notice that, the missing label results are better than

the corresponding noisy label results shown in Fig. 1(a). Take 20NG dataset as an example, we see that for the 60% missing label rate, the NMI is near 0.6 which still outperforms the *NCut* algorithm. However for the 60% noisy label rate, the NMI is around 0.3. Moreover, we find that for all the datasets, the clustering results of *DLSR* are better than the results of baseline *NCut* method for missing labels. This shows the initial label information is useful to improve the clustering results.

We also compare our algorithm with the semi-supervised clustering with side-information. We compare with one of the most popular methods, which is called Relevant Component Analysis (RCA) (Bar-hillel et al. 2005). We first perform PCA (Abdi and Williams 2010) to reduce the text data to 200 dimensional vectors and run RCA algorithm to get the Mahalanobis matrix for another dimensionality reduction problem. Then we perform Kmeans algorithm in the reduced space five times and output the best results. The results are shown in Fig. 1(c). It shows that our algorithm without dimensionality reduction is very competitive with the state-of-the-art algorithm.

Label Refinement for Other Clustering Algorithms

In this experiment, we use our algorithm to refine the output labels from other clustering algorithms. Particularly, we select some state-of-the-art clustering algorithms to generate the output labels to be refined. (1) Traditional Kmeans algorithm based on Euclidean distance (*Kmeans*). Since we make use of normalized *TF-IDF* feature as the input of all the algorithms, the clustering results of *Kmeans* is identical to Spherical Kmeans (Dhillon and Modha 2001). (2) Principal direction divisive partition (*PDDP*) (Boley 1998). (3) Normalized cut algorithm (*NCut*) (Shi and Malik 2000).

We first compare *Kmeans*, *PDDP*, and *NCut* algorithm by setting the ground truth class numbers. For our *DLSR* algorithm that uses both label similarity and dissimilarity, we first run *Kmeans* or *PDDP* as pre-clustering to generate initial labels. The pre-clustering cluster numbers are set to be 1,

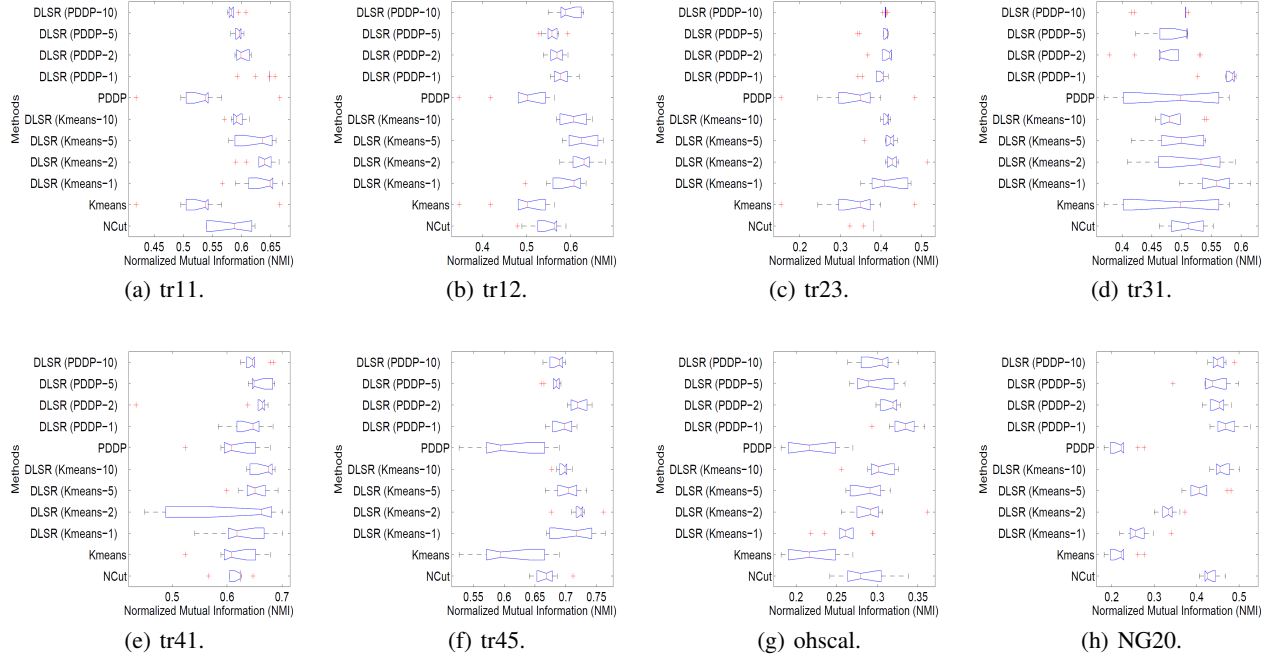


Figure 2: Clustering performance with different number of initial clusters on the eight datasets. The grouped boxes represent the results of different algorithms respectively. The whiskers are lines extending from each end of the box to show the extent of the rest of the data. Outliers are data with values beyond the ends of the whiskers, which are displayed by plus signs.

2, 5, and 10 times the true class number of each dataset. The results of NMI scores are shown in Fig. 2. From the results, we can see that the clustering results of *Kmeans* and *PDDP* are not good enough. On the contrary, our algorithm *DLSR* can significantly improve the clustering results of *Kmeans* and *PDDP* clustering. The results indicate that when the initial clustering results are not perfect in practice (e.g., results of *Kmeans* and *PDDP*), *DLSR* is able to refine the initial labels by combining data and label information. Moreover, although *DLSR* and *NCut* have the same essential property of graph cut, in most of the cases, our algorithm with different initial labels can outperform the original *NCut*. This means that by incorporating the initial labels generated from other algorithms, *DLSR* can jointly infer the better cluster labeling assignments by incorporating the good labels and discarding the noisy ones.

Impact of Label Dissimilarity on Label Refinement

Finally, to test the parameters that control the balance of similarity and dissimilarity in (10), we fix $a = 1$ and empirically set the value of b among $\{0, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$ where “0” represents that there is only label similarity involved. An example on the tr11 dataset is shown in Fig. 3 with 9 classes as ground truth. We use *Kmeans* as the pre-clustering algorithm to generate the initial cluster labels. The cluster number K is set as 0~10 times the class number. We see that the pre-clustering cluster number 2×9 shows the best results. Moreover, varying the value of b can obtain acceptable results in the range from 0 to 0.01.

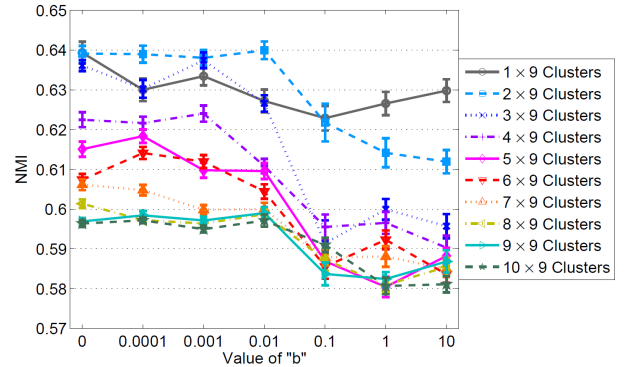


Figure 3: Tuning the value of b on tr11 dataset. *Kmeans* is used to pre-cluster the data. The pre-cluster numbers vary from 1 to 10 times the ground truth class number. The algorithm *DTSR* is then used to refine the labels of pre-clustering.

Conclusion and Future Work

We propose a label refinement algorithm to solve the noisy and missing labeled data problem. Instead of providing specific supervised model for different machine learning tasks, our algorithm could facilitate such learning tasks by refining the labels themselves in order to improve the performance of the particular task. Our algorithm uses both of the data content and label information, and benefits each other by jointly optimizing the smoothness function of labels over the con-

tent and label information. Experiments show that our label refinement algorithm can significantly generate refined labels from the noisy and missing labeled data. Moreover, it can also be used to improve the results of other clustering algorithms. To further improve the performance of our algorithm, it is possible to incorporate crowdsourcing (e.g., multiple labels from Amazon’s Mechanical Turks) into our algorithm in the future.

Acknowledgements

Yangqiu Song gratefully acknowledges the support by the Army Research Laboratory (ARL) under agreement W911NF-09-2-0053, by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20155, and by DARPA under agreement number FA8750-13-2-0008. The research is also partially supported by the National Natural Science Foundation of China (NSFC Grant No. 61472006), A Foundation for the Author of National Excellent Doctoral Dissertation of PR China (No. 201159), China National 973 program (No. 2014CB340304), and Hong Kong RGC Projects 621013, 620812, and 621211. Any opinions, findings, conclusions or recommendations are those of the authors and do not necessarily reflect the view of the agencies.

References

- Abdi, H., and Williams, L. J. 2010. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics* 2(4):433–459.
- Bar-hillel, A.; Hertz, T.; Shental, N.; and Weinshall, D. 2005. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research* 6(6):937–965.
- Boley, D. 1998. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery* 2(4):325–344.
- Chapelle, O.; Schölkopf, B.; and Zien, A., eds. 2006. *Semi-Supervised Learning*. Cambridge, MA: MIT Press.
- Chung, F. 1997. *Spectral Graph Theory*. Number 92 in CBMS Regional Conference Series in Mathematics. American Mathematical Society.
- Dhillon, I. S., and Modha, D. S. 2001. Concept decompositions for large sparse text data using clustering. *Machine Learning* 42(1–2):143–175.
- Goldberg, A.; Zhu, X.; and Wright, S. 2007. Dissimilarity in graph-based semi-supervised classification. In *AISTATS*, 155–162.
- Hersh, W.; Buckley, C.; Leone, T. J.; and Hickam, D. 1994. Ohsumed: An interactive retrieval evaluation and new large test collection for research. In *SIGIR*, 192–201.
- Ji, H., and Grishman, R. 2008. Refining event extraction through cross-document inference. In *ACL*, 254–262.
- Karypis, G. 2002. Cluto - a clustering toolkit. In *Technical Report*.
- Law, E.; Settles, B.; and Mitchell, T. 2010. Learning to tag from open vocabulary labels. In *ECML/PKDD*, 211–226.
- Li, Y.; Qi, Z.; Zhang, Z. M.; and Yang, M. 2013. Learning with limited and noisy tagging. In *ACM MM*, 957–966.
- McCallum, A. K. 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. In *Technical Report*.
- Natarajan, N.; Dhillon, I. S.; Ravikumar, P. D.; and Tewari, A. 2013. Learning with noisy labels. In *NIPS*, 1196–1204.
- Nettleton, D. F.; Orriols-Puig, A.; and Fornells, A. 2010. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review* 33(4):275–306.
- Ng, A. Y.; Jordan, M. I.; and Weiss, Y. 2001. On spectral clustering: Analysis and an algorithm. In *NIPS*, 849–856.
- Pal, C.; Mann, G.; and Minerich, R. 2007. Putting semantic information extraction on the map: noisy label models for fact extraction. In *AAAI Workshop on Information Integration on the Web*.
- Ramakrishnan, G.; Chitrapura, K. P.; Krishnapuram, R.; and Bhattacharyya, P. 2005. A model for handling approximate, noisy or incomplete labeling in text classification. In *ICML*, 681–688.
- Raykar, V. C.; Yu, S.; Zhao, L. H.; Valadez, G. H.; Florin, C.; Bogoni, L.; and Moy, L. 2010. Learning from crowds. *Journal of Machine Learning Research* 11:1297–1322.
- Sheng, V. S.; Provost, F.; and Ipeirotis, P. G. 2008. Get another label? improving data quality and data mining using multiple, noisy labelers. In *KDD*, 614–622.
- Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8):888–905.
- Snow, R.; O’Connor, B.; Jurafsky, D.; and Ng, A. Y. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *EMNLP*, 254–263.
- Whitehill, J.; Wu, T.-f.; Bergsma, J.; Movellan, J. R.; and Ruvolo, P. L. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *NIPS*, 2035–2043.
- Yang, T.; Mahdavi, M.; Jin, R.; Zhang, L.; and Zhou, Y. 2012. Multiple kernel learning from noisy labels by stochastic programming. In *ICML*, 233–240.
- Yu, S. X., and Shi, J. 2003. Multiclass spectral clustering. In *ICCV*, 313–319.
- Zelnik-manor, L., and Perona, P. 2004. Self-tuning spectral clustering. In *NIPS*, 1601–1608.
- Zhao, Y., and Karypis, G. 2001. Criterion functions for document clustering: experiments and analysis. In *Technical Report*.
- Zhong, S., and Ghosh, J. 2005. Generative model-based clustering of documents: a comparative study. *KAIS* 8:374–384.
- Zhou, D.; Platt, J. C.; Basu, S.; and Mao, Y. 2012. Learning from the wisdom of crowds by minimax entropy. In *NIPS*, 2204–2212.
- Zhu, X.; Wu, X.; and Chen, Q. 2003. Eliminating class noise in large datasets. In *ICML*, 920–927.