PAPER Special Section on Data Engineering and Information Management

Sentiment Classification in Under-Resourced Languages Using Graph-based Semi-supervised Learning Methods

Yong REN[†], **Nobuhiro KAJI**[††], **Naoki YOSHINAGA**[††], ***Nonmembers*, *and*** **Masaru KITSUREGAWA**[††,†††], ***Fellow***

SUMMARY In sentiment classification, conventional supervised approaches heavily rely on a large amount of linguistic resources, which are costly to obtain for under-resourced languages. To overcome this scarce resource problem, there exist several methods that exploit graph-based semi-supervised learning (SSL). However, fundamental issues such as controlling label propagation, choosing the initial seeds, selecting edges have barely been studied. Our evaluation on three real datasets demonstrates that manipulating the label propagating behavior and choosing labeled seeds appropriately play a critical role in adopting graph-based SSL approaches for this task.

key words: *sentiment classification, graph-based semi-supervised learning*

1. Introduction

Over the last decade, document-level sentiment classification has attracted much attention from NLP researchers. Potential applications include opinion mining and summarization [12]. Most of the existing methods [5, 8, 9, 13] locate sentiment classification as a supervised classification problem and train a reliable classifier from manually labeled data. The main disadvantage of those supervised approaches is that they demand a large amount of training data to achieve high accuracy.

Unfortunately, for some languages such as Chinese and Hindi, a sufficient amount of training data is not always available [22]. The annotation is known to be time consuming and requires substantial human labor by domain experts. Sentiment classification is therefore a quite challenging problem for such under-resourced languages.

Semi-supervised learning (SSL) algorithms are attractive approaches to address this problem. SSL methods can exploit labeled as well as unlabeled data. Unlike labeled data, unlabeled data are much easier to obtain. Thus, the demand for expensive labeled data can be highly relieved. As an important campaign, graph-based SSL methods (surveyed in [24]) have attracted a great deal of attention from research communities.

In this work, we focus on document-level sentiment classification under a minimally-supervised setting, where we only have a few labeled reviews given a priori. We explore two representative graph-based SSL algorithms (basic and state-of-the-art), label propagation (LP) [23] and modified adsorption (MAD) [18], to understand the behavior of graph-based SSL algorithms in this task setting. We empirically investigate the impact of controlling label propagation, choosing initial seeds, and pruning edges in exploiting graph-based SSL algorithms.

Experiments were carried out on three real datasets taken from different domains (hotel, notebook, and book) in Chinese.[*] We obtained the following findings through our thorough experiments.

- MAD outperformed LP in terms of the flexibility needed to alleviate the problem of (sentiment) polarity shift caused by high-degree vertices in a graph.
- Choosing initially-labeled seeds on the basis of their PageRank values or the number of neighbors can improve the performance in hotel and notebook domains.
- Pruning edges does not achieve a similar level of performance like in the choice of seeds.

The rest of this paper is organized as follows. In Sect. 2, related work is introduced. In Sect. 3, we explain the mechanisms of the SSL algorithms explored in our study. In Sect. 4, we evaluate those algorithms and investigate existing issues. In Sect. 5, we conclude this study and outline our future direction.

2. Work Related to SSL in Sentiment Classification

Modified adsorption (MAD), which is a graph-based SSL algorithm, is adopted in [17] to perform sentiment classification on Tweets. The authors leveraged characteristics of Twitter such as hashtags, emoticons, and follower-followee relationships to build a graph for MAD. Additionally, they studied the impact of labeled data as seeds on classification performance. However, their approach is not appropriate in an under-resourced scenario since the labeled data come from outside resources such as OpinionFinder.[**]

In [16], the authors designed a novel graph-based SSL algorithm to solve document-level sentiment classification. Their approach is based on a bi-partite graph composed of words and documents, which means the proposed method can assign sentiment polarity to both words and documents

Manuscript received July 13, 2013.
Manuscript revised October 29, 2013.

[†]Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-0133 Japan
[††]Institute of Industrial Science, The University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo, 153-8505 Japan
[†††]National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430 Japan

DOI: 10.1587/transinf.E97.D.1

[*]http://www.searchforum.org.cn/tansongbo/corpus-senti.htm
[**]http://www.cs.pitt.edu/mpqa/subj_lexicon.html

jointly. Their main focus is to encode prior lexical knowledge into the SSL paradigm with the help of regularized least squares.

Transductive SVM (TSVM) [7], which is a semi-supervised variant of SVM, is used in [4] to conduct the document-level sentiment classification task with the same under-resourced setting as in this study. Basically, their method is divided into three steps. First, they performed spectral clustering to identify unambiguous reviews. Second, they took advantage of active learning to label only the ambiguous reviews. Finally, they made use of the resulting labeled reviews and the remaining unlabeled reviews to train a TSVM classifier. Although they used an SSL-based approach to tackle the scarce resource problem in sentiment classification, they assume manual intervention in the active learning step.

In summary, these studies demand auxiliary resources or substantial human effort. To the best of our knowledge, this paper is the first study that explores graph-based SSL algorithms for sentiment classification in a real resource-scarce setting.

3. Graph-Based Semi-Supervised Learning

We explore semi-supervised sentiment classification, where a classifier is trained on both labeled data $\{(x_i, y_i)\}_{i=1}^{n_l}$ and unlabeled data $\{(x_j)\}_{j=n_l+1}^{n_l+n_u}$. In our work, x_i is represented by a feature vector, while y_i is the sentiment polarity of the review, *i.e.,* positive or negative. We assume there is no neutral category in this study, so in essence, our task is a binary classification problem.

Even if labeled data is costly to obtain in under-resource languages, we can usually compute a similarity between reviews to form a graph, where a vertex corresponds to a review and edges connect similar vertices. We thus can make use of graph-based SSL algorithms to perform sentiment classification. The choice of similarity measure is an open issue in using graph-based SSL algorithms, and we will later explore this in Sect. 4.3.2.

In this section, we explain two graph-based SSL algorithms, label propagation (LP) [23] and modified adsorption (MAD) [18], which we used in our work.

3.1 Formal Problem Setting

Graph-based SSL algorithms are formally given as an undirected graph, $G = (V, E, W)$, where $v \in V$ represents an example to be labeled, which corresponds to a review in our case, an edge $e = (a, b) \in E$ represents that the labels of the two vertices, a and b, are similar, and the weight $\mathbf{W}_{ab}$ represents the strength of the similarity. Since a vertex corresponds to an example, we have $n_l+n_u = |V|$. We use V_l and V_u to denote the set of vertices corresponding to the labeled and unlabeled examples, respectively.

The algorithm is also provided with initial label matrix $\mathbf{Y}$, where the row $\mathbf{Y}_v$ denotes the initial probability distribution over labels of the vertex v. For a vertex, $v \in V_l$, we have $\mathbf{Y}_{vy} = 1$ and $\mathbf{Y}_{vy'} = 0$ $(y' \neq y)$. For an unlabeled vertex, $v \in \mathbf{V}_u$, $\mathbf{Y}_v$ is set as a zero vector.

The goal of a graph-based SSL algorithm is to induce a probability distribution over labels of the vertices $\hat{\mathbf{Y}}$, where $\hat{\mathbf{Y}}_v$ represents the estimated probability distribution over labels of the vertex v.

3.2 Label Propagation

The first graph-based SSL algorithm we explore is LP. LP has a lot of advantages including a well-defined objective function and convergence property, and it has been successfully used in several NLP tasks [1, 10].

Mathematically, LP aims at minimizing the following objective function with respect to the labels that each vertex would own [14, 23].

$$\begin{aligned} &\frac{1}{2}\sum_{v,v' \in V} \mathbf{W}_{vv'}(\hat{\mathbf{Y}}_v - \hat{\mathbf{Y}}_{v'})^2 \\ &\textit{subject to } \hat{\mathbf{Y}}_v = \mathbf{Y}_v (v \in V_l) \end{aligned} \tag{1}$$

Eq. 1, which is sometimes referred to as energy or smoothness, is the common objective function in the graph-based SSL method. Intuitively, LP can be interpreted as assigning the same labels to vertices that are connected by edges with large weights while fixing the labels of the vertices corresponding to labeled data.

It is not difficult to verify that the solution of Eq. 1 satisfies the following stationary conditions.

$$\begin{aligned} \hat{\mathbf{Y}}_v &= \mathbf{Y}_v \ (v \in V_l) \\ \hat{\mathbf{Y}}_v &= \frac{1}{d_v}\sum_{v'} \mathbf{W}_{v'v}\hat{\mathbf{Y}}_{v'} \ (v \in V_u) \\ \textit{where } d_{v'} &= \sum_{v} \mathbf{W}_{vv'} \end{aligned} \tag{2}$$

Eq. 2 can be further transformed into matrix form $\hat{\mathbf{Y}}_v = \mathbf{T}\hat{\mathbf{Y}}_v$, where $\mathbf{T} = \mathbf{D}^{-1}\mathbf{W}$ and $\mathbf{D} = \mathrm{diag}(d_v)$. Then, we can seek the $\hat{\mathbf{Y}}_v (v \in V_u)$ that satisfies Eq. 2 in an iterative manner.

Algorithm 1 depicts LP in detail. In the initiation section (line 1 in Algorithm 1), it first initializes the label matrix $\hat{\mathbf{Y}}$. After the initialization, a new matrix, $\mathbf{T}$, is built through transforming the weight matrix $\mathbf{W}$ (line 2). Then, LP enters the learning phase (from line 3 to line 6) and propagates

Algorithm 1: Label Propagation

input: **Similarity graph**: $G = \{V, E, W\}$
Initial label matrix: Y

1 Initialize label matrix $\hat{\mathbf{Y}}$ by using seed examples
2 $\mathbf{T} = \mathbf{D^{-1}W}$
3 **while** *$\hat{Y}$ is not convergent* **do**
4 $\quad \hat{\mathbf{Y}} = \mathbf{T}\hat{\mathbf{Y}}$
5 $\quad \hat{\mathbf{Y}}_v = \mathbf{Y}_v(v \in V_l)$ # Clamp the seed examples in Y to their original values
6 **end**
Output: $\hat{\mathbf{Y}}$

labels through the graph (line 4). In essence, it is an iterative matrix computation. At the end of each iteration, the seeds are re-adjusted to the original value (line 5). When the matrix $\hat{\mathbf{Y}}$ converges, the propagation terminates.

Note that LP will suffer from densely connected components in the similarity graph, especially when the weights of edges are not reliable [21]. We guess that high-degree vertices are the origin of that problem and explore the way adsorption tackles the problem in the following subsection.

3.3 Adsorption

In this section, we explain the adsorption algorithm as it provides the basis of MAD, which is used in our experiment. Note that adsorption itself is not used in our experiment.

Conventional graph-based SSL algorithms such as LP suffer from *topic drift* caused by high degree vertices [2]. Adsorption handles this problem by controlling the label propagation process one the basis of three actions. First, it abandons the propagation process at vertex v with probability p_v^{abnd}. Second, it simply returns the initial label distribution $\mathbf{Y}_v$ at vertex v with probability p_v^{inj}. Note that $p_v^{inj} = 0$ for $v \in V_u$. Finally, it continues to propagate label information with probability p_v^{cont}. The resulting label distribution $\hat{\mathbf{Y}}$ is given as

$$\hat{\mathbf{Y}}_v = p_v^{inj} \times \mathbf{Y}_v + p_v^{cont} \times \sum_{v':(v',v)\in E} \Pr[v'|v]\hat{Y}_{v'} + p_v^{abnd} \times \mathbf{r}$$
$$where\ \Pr[v'|v] = \frac{\mathbf{W}_{v'v}}{\sum_{u:(u,v)\in E} \mathbf{W}_{uv}} \quad (3)$$

Algorithm 2 illustrates the adsorption algorithm. We can clearly see the difference between LP and adsorption from Algorithm 2. First, the labels of vertices $v \in V_l$ are allowed to be re-adjusted, unlike in LP. The main motivation of this strategy is to deal with noise or initial unreliable labels. Furthermore, adsorption brings inject (p_v^{inj}), continue (p_v^{cont}), and abandon probabilities (p_v^{abnd}) into the label diffusing process (line 5). Therefore, adsorption could be adapted to diverse graphs in a more flexible way at the price of learning complexity. Finally, a dummy vector, $\mathbf{r}$ (line 5), is added so that adsorption can assign an arbitrary label to the corresponding vertex when the label propagation is abandoned.

Algorithm 2: Adsorption

input: **Similarity graph**: $G = \{V, E, W\}$
Initial label matrix: Y
Probabilities: p_v^{inj}, p_v^{cont}, p_v^{abnd} for $v \in V$

1 $\hat{\mathbf{Y}}_v = \mathbf{Y}_v$ for $v \in V$
2 **while** $\hat{Y}_v$ *is not convergent* **do**
3 $\quad \mathbf{D}_v = \frac{\sum_u \mathbf{W}_{uv}\hat{\mathbf{Y}}_v}{\sum_u \mathbf{W}_{uv}}$ for $v \in V$
4 $\quad$ **for** $v \in V$ **do**
5 $\quad\quad \hat{\mathbf{Y}}_v = p_v^{inj} \times \mathbf{Y}_v + p_v^{cont} \times \mathbf{D}_v + p_v^{abnd} \times \mathbf{r}$
6 $\quad$ **end**
7 **end**

Output: $\hat{\mathbf{Y}}_v$

The values of probability vectors denoted by p_v^{inj}, p_v^{cont}, and p_v^{abnd} play a crucial role. While we manually adjusted those hyper-parameters to investigate the sensitivity, we here introduce an automatic approach proposed in [18] for interested readers.

Each vertex v has three probability values: p_v^{inj}, p_v^{cont}, and p_v^{abnd}. The value of inject probability p_v^{inj} (for the labeled vertex) is dependent on the label entropy. Since high entropy means more uncertainty, MAD prefers to use the pre-defined labels when the entropy is high. The setting of the continue probability p_v^{cont} for the vertex v is based on the number of neighbors it has. The intuition behind this setting is that the fewer the neighbors of the vertex v, the more label information they contain on the vertex v. Therefore, the vertex v should be encouraged to learn the label from its connections and vice versa. Specifically, the whole process can be formulated compactly in [18].

We first define the entropy of the transition probability.

$$H(v) = -\sum_{u:(u,v)\in E} \Pr[u|v] \log \Pr[u|v]$$
$$\Pr[u|v] = \frac{\mathbf{W}_{uv}}{\sum_u \mathbf{W}_{uw}} \quad (4)$$

By using the entropy, we define two values, g_v and h_v, on the basis of which p_v^{cont} and p_v^{inj} are defined.

$$f(x) = \frac{log\beta}{\log(\beta + e^x)} \quad (5)$$

The function $f(x)$ defined in Eq. 5 and is a monotonically decreasing function.

$$g_v = f(H_v) \quad (6)$$
$$h_v = (1 - g_v)\sqrt{H_v} \quad (7)$$

Obviously, g_v and h_v are respectively proportional and inversely proportional to the entropy defined in Eq. 5. By using g_v and h_v, the probability p_v^{cont} and p_v^{inj} is defined as

$$p_v^{cont} = \frac{g_v}{\max(g_v + h_v, 1)} \quad (8)$$
$$p_v^{inj} = \frac{h_v}{\max(g_v + h_v, 1)} \quad (9)$$
$$p_v^{abnd} = 1 - p_v^{inj} - p_v^{cont} \quad (10)$$

3.4 Modified Adsorption (MAD)

Despite the advantage adsorption owns, as pointed out in [18], there is no objective function in adsorption. Modified adsorption (MAD) alters the original adsorption algorithm so that it can own an objective function, and then we can gain the global optimal solution through optimization methodologies. In the following, we depict the formalization of the final objective function in MAD.[†]

There are three factors that are considered in MAD: the

[†]Interested readers may refer to the detailed derivation in [18].

labels predicted and the priori for the seeds should be consistent (Eq. 11), similar vertices bear the same labels (Eq. 12), and regularization should be performed (Eq. 13).

The purpose of Eq. 11 is to keep the consistency between the predicative results ($\hat{\mathbf{Y}}_l$) and the corresponding labeled instances $\mathbf{Y}_l$.

$$\sum_v p_v^{inj} \sum_l (\mathbf{Y}_{vl} - \hat{\mathbf{Y}}_{vl})^2 = \sum_l (\mathbf{Y}_l - \hat{\mathbf{Y}}_l)^{\mathrm{T}} \mathbf{S} (\mathbf{Y}_l - \hat{\mathbf{Y}}_l) \quad (11)$$

where matrix $\mathbf{S}$ is diagonal ($\mathbf{S} = diag(p_v^{inj})$)

Next, the similarity matrix is transformed with $\mathbf{W}'_{vu} = p_v^{cond} \times \mathbf{W}_{vu}$. Thus, vertex u is not similar to vertex v, which has a large-degree (the value of p_v^{cond} is low).

$$\begin{aligned} \sum_{v,u} \mathbf{W}'_{v\mu} \left\| \hat{\mathbf{Y}}_v - \hat{\mathbf{Y}}_u \right\|_2^2 &= \sum_l \sum_{v,u} W'_{vu} (\hat{\mathbf{Y}}_{vl} - \hat{\mathbf{Y}}_{ul}) \\ &= \sum_l \hat{\mathbf{Y}}_l^{\mathrm{T}} \mathbf{L} \mathbf{Y}_l \end{aligned} \quad (12)$$

where, $\mathbf{L} = \mathbf{D} + \overline{\mathbf{D}} - \mathbf{T} - \mathbf{W}'$ and $\mathbf{D}$, $\overline{\mathbf{D}}$ are $n \times n$ diagonal matrices with $D_{vv} = \sum_u \mathbf{W}'_{uv}$, $\overline{\mathbf{D}}_{vv} = \sum_u \mathbf{W}'_{vu}$. The purpose of Eq. 12 is to distribute labels smoothly across the graph.

Eq. 13 takes the responsibility of regularization.

$$\sum_{vl} (\hat{\mathbf{Y}}_{vl} - \mathbf{R}_{vl})^2 = \sum_l \left\| \hat{\mathbf{Y}}_l - \mathbf{R}_l \right\|_2^2 \quad (13)$$

The elements of the last column in matrix $\mathbf{R}$ are set to the corresponding $p_v^{abnd} \times \mathbf{r}$, while the elements of the other columns are 0.

The objective function is constructed by combining the above three equations:

$$\begin{aligned} C(\hat{\mathbf{Y}}) = \sum_l [&\mu_1 (\mathbf{Y}_l - \hat{\mathbf{Y}}_l)^T \mathbf{S} (\mathbf{Y}_l - \hat{\mathbf{Y}}_l) \\ &+ \mu_2 \hat{\mathbf{Y}}_l^T \mathbf{L} \mathbf{Y}_l + \mu_3 \left\| \hat{\mathbf{Y}}_l - \mathbf{R}_l \right\|_2^2] \end{aligned} \quad (14)$$

Algorithm 3 depicts the MAD algorithm. Three hyper-parameters, μ_1, μ_2, and μ_3 (line 2), are used to emphasize the importance of related constraints. An efficient way to compute the optimal minima was proposed [18].

4. Evaluation and Discussion

We start by introducing datasets followed by an explanation of pre-processing them for evaluation. We then demonstrate the results of our evaluation, which include a performance comparison of SVM, LP, and MAD and the impact of similarity measure, tuning hyper-parameters, pruning unreliable edges, and selecting seeds.

Algorithm 3: Modified Adsorption

input: **Similarity graph**: $G = \{V, E, W\}$
Initial label matrix: Y
Probabilities: p_v^{inj}, p_v^{cont}, p_v^{abnd} for $v \in V$

1 $\hat{\mathbf{Y}}_{\mathbf{v}} = \mathbf{Y}_v$ for $v \in V$
2 $\mathbf{M}_{vv} = \mu_1 \times p_v^{inj} + \mu_2 \times \sum_{u \neq v} (p_v^{cont} \mathbf{W}_{vu} + p_u^{cont} \mathbf{W}_{uv}) + \mu_3$
3 **while** $\hat{Y}_v$ *is not convergent* **do**
4 $\mathbf{D}_v = \sum_u (p_v^{cont} \mathbf{W}_{vu} + p_u^{cont} \mathbf{W}_{uv}) \hat{\mathbf{Y}}_u$
5 **for** $v \in V$ **do**
6 $\hat{\mathbf{Y}}_{\mathbf{v}} = \frac{1}{\mathbf{M}_{vv}} (\mu_1 \times p_v^{inj} \times \mathbf{Y}_v + \mu_2 \times \mathbf{D}_v + \mu_3 \times p_v^{abnd} \times \mathbf{r})$
7 **end**
8 **end**

Output: $\hat{\mathbf{Y}}_v$

4.1 Setting

The datasets we used in the following experiments were from ChnSentiCorp (de-duplicate version).[†] They consist of reviews from three different domains: notebook, hotel, and book (around 4000 reviews in each domain). Each review is manually labeled with sentiment polarity (positive or negative), and each of the three sets of reviews is balanced in terms of sentiment polarity.

In each domain, we randomly selected 300 reviews as test data. The test data were balanced in terms of sentiment polarity (150 positive and 150 negative reviews) so that the random baseline achieved a classification accuracy of 0.5. We also selected labeled data at random. The number of reviews in the training data (balanced in terms of sentiment polarity) was varied from 20 to 300 to investigate the effect of the amount of supervision. The remaining reviews were used as unlabeled data for semi-supervised learning.

Because the accuracy of the classifiers could depend on the choice of labeled reviews, especially when we choose a small number reviews to label (here 20, minimum), we ran the experiments ten times by randomly choosing reviews to be labeled. We report the average of the classification accuracy as the final result.

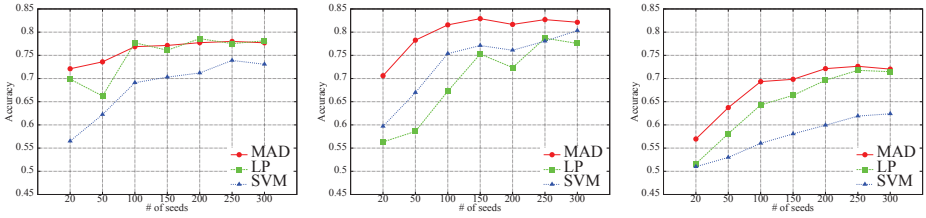
To explore the advantage of graph-based SSL algorithms over supervised counterparts, we used SVM [20], which is a widely-used supervised classification algorithm, as a baseline.

We used SVMlight[††] as the implementation of SVM in our experiments, while we adopted Junto[†††] as the implementation of LP and MAD.

4.2 Pre-processing

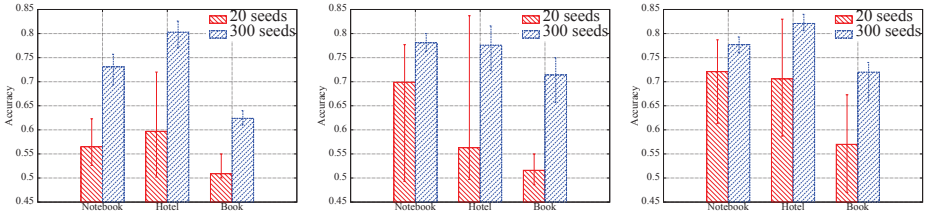
In this section, we introduce the features used to represent each review. In this study, each review is represented as a bag-of-features, and they are used to measure the similarity in building a graph for graph-based SSL algorithms while it is also an input to SVM. In [15], we investigated the topic of sentiment features exhaustively and concluded that specific phrases extracted by manually-tailored POS patterns are the best option because they capture the proper context related to the sentiment expressed. In this work, we follow [15] to choose phrases with specified POS patterns as sentiment features (prior to extracting the specific phrases, Stanford Word Segmenter[††††] and Log-linear Part-Of-Speech Tagger

[†] http://www.searchforum.org.cn/tansongbo/corpus-senti.htm
[††] http://svmlight.joachims.org/
[†††] https://github.com/parthatalukdar/junto
[††††] http://nlp.stanford.edu/software/segmenter.shtml



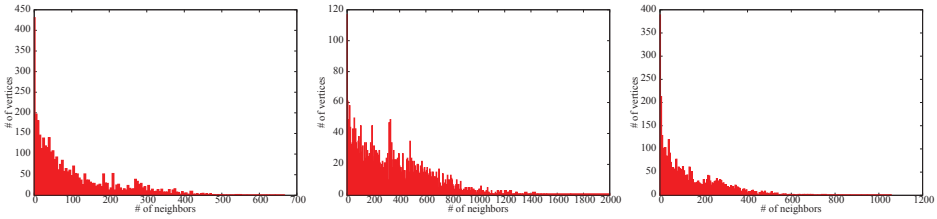
(a) Notebook reviews (b) Hotel reviews (c) Book reviews

Fig. 1 Performance comparison



(a) SVM (b) LP (c) MAD

Fig. 2 Performance deviation



(a) Notebook reviews (b) Hotel reviews (c) Book reviews

Fig. 4 Vertex degree distribution

Table 1 POS patterns and example sentiment features that match them

POS pattern	Sentiment features
AD VA	真的不错(really not bad) 太困难 (too difficult)
AD VV	很生气 (very angry) 不犹豫 (do not hesitate)
AD JJ	太慢 (too slow) 那么简单 (so simple)
NN JJ	环境一流 (environment excellent) 设施旧(facilities old)
NN VA	态度不错 (attitude OK) 语言简洁 (language concise)

Table 2 Reviews and their sentiment phrases

Reviews	Sentimental features extracted
服务天都不错，吃的很好。 (Service attitude is OK. The food is delicious.)	态度不错 很好 (Attitude is OK, delicious)
房间很小，很冷，不满意！ (The room is very small and cold. Unsatisfied!)	很小 很冷 不满意 (Very small, very cold, unsatisfied)

for Chinese[†††††] are used to pre-process each review).

Table 1 lists the five POS patterns [15] used to extract phrases along with corresponding ones extracted by them, and Table 2 lists positive and negative reviews in the hotel domain of the dataset along with corresponding feature representations.

[†††††]http://nlp.stanford.edu/software/tagger.shtml

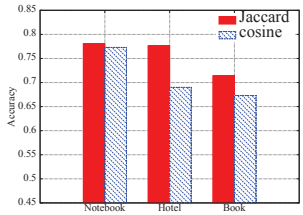


Fig. 3 Comparison between Jaccard and cosine similarity

4.3 Results

The followings are the objectives of our evaluation.

- Compare the performance of LP and MAD with SVM in the document-level sentiment classification task when limited training data are available (Sect. 4.3.1).
- Show the influence of the similarity measure and hyper-parameters (Sects. 4.3.2 and 4.3.3).
- Investigate the impact of selecting seeds and pruning edges (Sects. 4.3.4 and 4.3.5).

4.3.1 Impact of the Number of Seeds

In this section, we used the Jaccard similarity coefficient [6] to compute the similarity between two reviews. A comparison of classification performance is shown in Fig. 1, where the vertical axis indicates the classification accuracy, and the horizontal axis indicates the number of labeled reviews. Here, we set the hyper-parameters in MAD as the default values. The impact of these hyper-parameters is shown in Sect. 4.3.3.

The performance of LP was bad when the number of labeled seeds was very small (especially, 20-50). A possible culprit is the noise structure of similarity graphs. Some commonly-extracted phrases create many undesired edges that connect positive and negative instances, which causes a sentiment polarity drift during the process of label propagation. Note that the "mis-connection" phenomenon among instances is common in building similarity graphs for graph-based SSL algorithms. How to take effective measures to tackle this phenomenon is still an open question.

As the improved version of LP, MAD can outperform LP in most cases, especially when the size of available labeled seeds is limited. We credit this with the capability of MAD to tackle noise in the graph with flexibility. After incorporating hyper-parameters, the role of labeled data is emphasized properly, and at the same time, the label propagating behavior for especially the high-degree vertices gets appropriate control. The lesson learnt here is when we cannot construct desirable graph, taking the strategy to control label distribution is helpful.

Not surprisingly, we can see that, with the increase of labeled instances, the performances of all the methods are improved. For MAD and LP, when more labeled data are available, unlabeled vertices could get more reliable sources so that MAD and LP could become more confident to decide the label one specific vertex belongs to. For SVM, the increase of labeled instances means that more training data are available, so it can locate a more accurate hyperplane.

Finally, all of those approaches do not perform well in the book domain. Because book reviews cover various aspects, including the story, the writing style of the author, the characters appearing in the book, and even the reputation of the publisher. It is common for the sentiments of these aspects to not be consistent. Turney [19] reported similar findings for movie reviews.

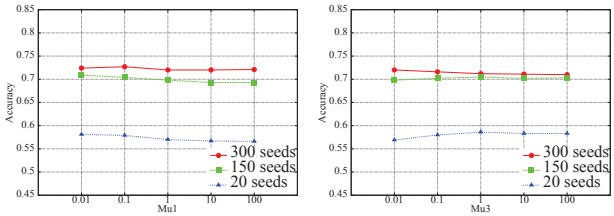
We show the performance deviation in Fig. 2, where we can clearly find that the accuracy was highly sensitive to the choice of seeds when the number of seeds was small (here 20). We explored strategies such as pruning unreliable edges (Sect. 4.3.4) and selecting seeds (Sect. 4.3.5) to relieve the performance sensitivity.

4.3.2 Impact of Similarity Measure in Building a Graph

The similarity measure is one of the important factors that affects the accuracy of graph-based SSL algorithms. When the similarity score between two reviews is not zero, we build an edge between them. A performance comparison (300 labeled seeds) between the two common similarity measures, Jaccard similarity coefficient and cosine similarity (with TF-IDF feature weighting), is shown in Fig. 3. We can observe that Jaccard is a better option. Interested readers may refer to [15] to see a comparison among various similarity measures.

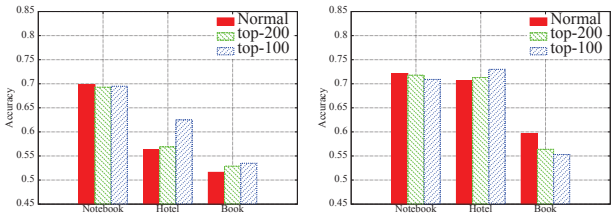
Table 3 Similarity graphs statistics

Domain	No. of vertices	No. of edges	LCS size	Max degree
Notebook	3936	209,203	3936	658
Hotel	3814	667,198	3814	1878
Book	3831	267,678	3831	1059



(a) Impact of μ_1 (b) Impact of μ_3

Fig. 5 Impact of μ_1 and μ_3 for books



(a) LP (b) MAD

Fig. 6 Impact of edge selection

Table 3 contains the statistics on the similarity graphs in the three different domains. We can find the largest connected component included all the vertices in the similarity graph, which allowed not only LP but also MAD to label all the reviews. Fig. 4 shows the degree distribution of the vertices. We could observe that the number of neighbors of vertices in the hotel domain was much larger than those in the other two domains. When we launch an edge pruning task (such as in our case in Sect. 4.3.4), we should take the degree distribution into consideration.

4.3.3 Impact of Tuning Hyper-parameters

In MAD, we are especially interested in the impact of inject probabilities and abandon probabilities because they enhance the label diffusing behavior. At present, we therefore set the value of hyper-parameter μ_2 to its default value.

First, as we depicted in Sect. 3.3, the labeled instances are not adjusted to the original states in MAD. When the similarity graph includes noisy edges that we usually confront, we need to put a high value to μ_1 to keep the consis-

tency between the original labels and labels predicted for labeled seeds. We could guess that in a noisy similarity graph, we must ensure the correctness of labeled data (leaders, borrowing a wording in [18]) so that the classification performance (whole world) cannot degenerate (go out of control).

Second, when the number of labeled data is not sufficient, the labels predicted are not reliable. Worse, high-degree vertices will propagate wrong labels to the neighbors (see Sect. 4.2). In such a kind of circumstance, the value of μ_3 should be high so that vertices become conservative in propagating labels to their neighbors.

Finally, we also found that these controlling behaviors are not highly sensitive to the value of related hyper-parameters. Here, we only show the influence of related hyper-parameters in the book domain since a similar tendency was observed in the notebook and book domains. Figure 5 shows the influence of μ_1 and μ_3 in MAD. We first set μ_3 as the default value and adjusted the value of μ_1; then, we kept μ_1 as the default value and shifted the process. We could only observe a slight difference in performance.

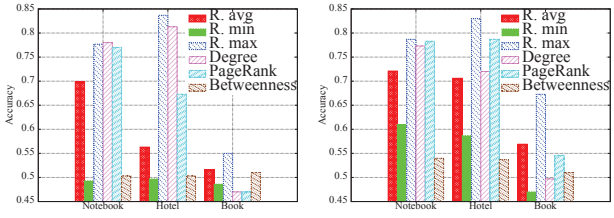
4.3.4 Impact of Pruning Unreliable Edges

The critical assumption behind graph-based SSL algorithms is that two vertices with a high weight connection tend to bear the same label. Therefore, pruning unreliable edges is a straightforward way of improving the performance.

We thus explore the effectiveness of the method of selecting proper edges. We explored the following strategy to prune unreliable edges. First, given one vertex, we rank its neighboring edges in accordance with the weight, and then, we keep the top-N edges. As we showed in Fig. 4 in Sect. 4.2, the degree distribution varied in the different domains. Hence it is obvious that we had more candidates in the hotel domain than in the other two domains. Taking this into account, we left top-100 and top-200 edges for each vertex. The rational is that we wanted to simultaneously choose edges and keep good connectivity in the graph. The number of labeled seeds was set to 20 (10 in each class), and we again ran experiments ten times while varying randomly-chosen initial seeds and averaging the obtained accuracy. We present the impact of pruning unreliable edges in Fig. 6. Compared with the case where edge selection was not performed, there was moderate improvement. However, we could not get a clear and consistent tendency. We also conducted a statistical significance test (t-test) on the results, and we observed that all the p-values were above 0.2. We conclude that pruning unreliable edges is not an effective strategy to improve the performance.

4.3.5 Impact of Selecting Seeds

In a scarce resource setting, we usually do not have any labeled seeds, so we need to determine the examples to label. It is known that the importance of vertices in a given graph is different, so one intuitive labeling strategy is to choose vertices with high importance metric as labeled seeds. In this study, we explored three criteria for selecting seeds: the number of degrees, the PageRank [11] value, and the betweenness centrality [3]. We compared them with the randomly chosen seeds. Note that we also need to balance the sentiment polarity of seeds selected as we did in Sect. 4.3.1; otherwise, unbalanced classification would occur. To meet this requirement, we can scan the vertices (measured by using the number of degrees, the PageRank value and the betweenness centrality, respectively) and annotate them until we accumulate a certain number (10 in our case) of labeled seeds for each class.



(a) LP (b) MAD

Fig. 7 Impact of seed selection

Figure 7 shows the influence of seed selection, where the average value, the minimum, and the maximum of performance with randomly selected seeds are denoted as "R. avg," "R. min," and "R. max," respectively. We can conclude that selecting nodes with a high degree and PageRank value as seeds are the best choices for LP and MAD, respectively. The results are comparable with the best ones ("R. Max"). However, the book domain is an exception. Choosing seeds randomly may be the optimal option when the connection in the graph does not reflect the class similarity well. Vertices with high betweenness centrality are critical to connect the other vertices in the graph, but they are not good choice as seeds (the sources for labels).

In essence, selecting initially-labeled seeds is an easily controlled way to improve the performance when we merely hope to label a small number of data. We can make use of existing metrics such as PageRank value and the number of degrees to realize this purpose without modifying the topology of graphs, which may cause cascading changes to the graph-based learning behavior.

5. Conclusion and Future Work

In this paper, we empirically investigated the usefulness of two graph-based SSL algorithms, LP and MAD, to solve document-level sentiment classification for under-resourced languages. In particular, we found that choosing initially-labeled vertices in accordance with their degree and PageRank score can improve the performance. However, pruning unreliable edges will make things more difficult to predict. We believe that other people who are interested in this field can benefit from our empirical findings.

As future work, first, we will attempt to use a sophisticated approach to induce better sentiment features. We consider such elaborated features improve the classification

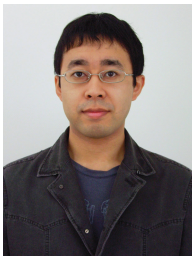
performance, especially in the book domain. We also plan to exploit a much larger amount of unlabeled data to fully take advantage of SSL algorithms. Finally, we are interested in applying SSL algorithms to sentiment classification in domains, such as blogs, tweets, and so forth.

References

[1] A. Alexandrescu and K. Kirchhoff. Graph-based learning for statistical machine translation. In *Proc. HLT-NAACL*, pages 119–127, 2009.
[2] S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proc. WWW*, pages 895–904, 2008.
[3] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(163), 2001.
[4] S. Dasgupta and V. Ng. Mine the easy, classify the hard: a semi-supervised approach to automatic sentiment classification. In *Proc. ACL-IJCNLP*, pages 701–709, 2009.
[5] M. Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proc. COLING*, pages 841–847, 2005.
[6] P. Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 11(2):37–50, Feb. 1912.
[7] T. Joachims. Transductive inference for text classification using support vector machines. In *Proc. ICML*, pages 200–209, 1999.
[8] S. Matsumoto, H. Takamura, and M. Okumura. Sentiment classification using word sub-sequences and dependency sub-trees. In *Proc. PAKDD*, pages 301–311, 2005.
[9] T. Mullen and N. Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proc. EMNLP*, pages 412–418, 2004.
[10] Z.-Y. Niu, D.-H. Ji, and C. L. Tan. Word sense disambiguation using label propagation based semi-supervised learning. In *Proc. ACL*, pages 395–402, 2005.
[11] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. In *Proc. WWW*, pages 161–172, 1999.
[12] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2:1–135, January 2008.
[13] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proc. EMNLP*, pages 79–86, 2002.
[14] D. Rao and D. Ravichandran. Semi-supervised polarity lexicon induction. In *Proc. EACL*, pages 675–682, 2009.
[15] Y. Ren, N. Kaji, N. Yoshinaga, M. Toyoda, and M. Kitsuregawa. Sentiment classification in resource-scarce languages by using label propagation. In *Proc. PACLIC*, pages 420–429, 2011.
[16] V. Sindhwani and P. Melville. Document-word co-regularization for semi-supervised sentiment analysis. In *Proc. ICDM*, pages 1025–1030, 2008.
[17] M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proc. EMNLP*, pages 53–63, 2011.
[18] P. P. Talukdar and K. Crammer. New regularized algorithms for transductive learning. In *Proc. ECML-PKDD*, pages 442–457, 2009.
[19] P. D. Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proc. ACL*, pages 417–424, 2002.
[20] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995.
[21] L. Velikovich, S. Blair-Goldensohn, K. Hannan, and R. McDonald. The viability of web-derived polarity lexicons. In *Proc. HLT-NAACL*, pages 777–785, 2010.
[22] X. Wan. Co-training for cross-lingual sentiment classification. In *Proc. ACL-IJCNLP*, pages 235–243, 2009.
[23] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002.
[24] X. Zhu and A. B. Goldberg. *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers, 2009.

[Yong Ren] is currently a Ph.D. candidate of Graduate School of Information Science and Technology, The University of Tokyo, Japan. He received his B.Sc. degree and M.Sc. degree in Computer Science from Liaoning University of Technology and Communication University of China, respectively. His research interests include natural language processing and machine learning.

[Nobuhiro Kaji] is currently a project associate professor of the institute of industrial science, the university of Tokyo, Japan. He received his Ph.D. degree in information science and technology, from the university of Tokyo, Japan in 2005. He worked at the institute of industrial science, the university of Tokyo, as a research associate and project assistant professor from 2005 to 2006 and from 2006 to 2012. His research interests include natural language processing and machine learning.



[Naoki Yoshinaga] is currently a Project Associate Professor of the Institute of Industrial Science, the University of Tokyo, Japan. He received his B.Sc. and M.Sc. degrees in Information Science, and Ph.D. degree in Information Science and Technology, from the University of Tokyo, Japan in 2000, 2002, 2005. He was a JSPS research fellow from 2002 to 2005 (DC1) and from 2005 to 2008 (PD). He worked at the Institute of Industrial Science, the University of Tokyo, as a Project Researcher and a Project Assistant Professor from 2008 to 2012. His research interests include computational linguistics and machine learning.

[Masaru Kitsuregawa] is the Director General at National Institute of Informatics (NII) in Japan and is also a Professor and the Executive Director for Earth Observation Data Integration and Fusion Research Initiative (EDITORIA), at the University of Tokyo. He is also the president of Information Processing Society of Japan since June 2013. He received the Ph.D. degree in information engineering in 1983 from the University of Tokyo. His research interests include high performance database engineering, big data system. He is running earth environmental digital earth of more than 10PB. He serves as a chair of the steering committee of IEEE ICDE (Int._Conf. on Data Engineering), and served a trustee of the VLDB Endowment. He is a recipient of the ACM SIGMOD E. F. Codd Innovation Award in 2009. He was serving as a science advisor to the Ministry of Education, Culture, Sports, Science and Technology in Japan. He is a fellow of the ACM and IEEE.