Sentiment Classification in Resource-Scarce Languages by using Label Propagation

Yong Ren[a], Nobuhiro Kaji[b], Naoki Yoshinaga[b], Masashi Toyoda[b], and Masaru Kitsuregawa[b]

[a] Graduate School of Information Science and Technology, University of Tokyo, Japan
renyong@tkl.iis.u-tokyo.ac.jp
[b] Institute of Industrial Science, University of Tokyo, Japan
{kaji,ynaga,toyoda,kitsure}@tkl.iis.u-tokyo.ac.jp

Abstract. With the advent of consumer generated media (e.g., Amazon reviews, Twitter, etc.), sentiment classification becomes a heated topic. Previous work heavily relies on a large amount of linguistic resources, which are difficult to obtain in resource-scarce languages. To overcome this problem, we investigate the usefulness of label propagation, which is a graph-based semi-supervised learning method. Extensive experimental evaluation on three real datasets demonstrated that label propagation performs more stable than support vector machines (SVMs) and transductive support vector machines (TSVMs) in a document-level sentiment classification task for resource-scarce languages (Chinese in our case).

Keywords: sentiment classification, label propagation, semi-supervised learning

1 Introduction

Over the last decade, document-level sentiment classification has attracted much attention from NLP researchers; its potential applications include opinion summarization and opinion mining (Pang and Lee, 2008). Most of the existing methods locate sentiment classification as a supervised classification problem and train a reliable classifier from a large amount of labeled data (Pang *et al.*, 2002; Mullen and Collier, 2004; Matsumoto *et al.*, 2005; Gamon, 2005). The main disadvantage of such supervised approaches is that it is quite expensive in both time and labor to annotate a large amount of training data.

Unfortunately, in some languages such as Chinese and Hindi, a sufficient amount of training data is not always available. Sentiment classification becomes a quite challenging problem for such resource-scarce languages. While some studies have tackled this problem (Wan, 2009; Dasgupta and Ng, 2009), they still require substantial human efforts or specific linguistic resources that are only available in particular languages as we will see later in Section 2. We therefore want to develop a low-cost, general method that can be readily applicable to sentiment classification in any languages.

In this paper, we explore the use of label propagation (LP) (Zhu and Ghahramani, 2002) in building a document-level sentiment classifier under a minimally-supervised setting, where we have only a small number of labeled reviews other than the target reviews that we want to classify. Having a similarity graph whose vertices are labeled or unlabeled instances (reviews, here) and edges represent the similarity between the vertices, LP infers the label (sentiment polarity) of unlabeled (target) reviews based on labeled reviews that are similar to the target reviews. The key in applying LP to document-level sentiment classification is therefore in the way to represent reviews on the graph and to define the similarity between the them. We thus investigate the impact of the review representation and the similarity measure on the classification performance.

The main contributions of our work are summarized as follows:

25th Pacific Asia Conference on Language, Information and Computation, pages 420–429

- We evaluate LP on document-level sentiment classification in a resource-scarce language. Our method can be applied to any languages in which a small number of labeled reviews are available.
- We run LP with different review representations[1] (content words, phrases, and adjectives), and various similarity measures (dice coefficient, overlap coefficient, Jaccard, cosine similarity) (Manning and Schütze, 1999). We thereby reveal their impact on the classification performance.
- We compare our method with support vector machines (Vapnik, 1995) and transductive support vector machines (Joachims, 1999), and demonstrate the stability of the classification performance of our method in this task.

The rest of this paper is organized as follows: Section 2 introduces related work, Section 3 explains LP algorithm in detail. Section 4 evaluates our method. Section 5 concludes this study and discusses future direction.

2 Related Work

Several researchers attempted to solve a sentiment classification task in languages without abundant training instances (Dasgupta and Ng, 2009; Wan, 2009; Yanyan Zhao and Liu, 2010). In what follows, we briefly introduce those studies.

Dasgupta and Ng (2009) used transductive SVM (Joachims, 1999) to exploit unlabeled reviews in a document-level sentiment classification task. Basically their work is divided into three steps: firstly they perform spectral clustering to identify unambiguous reviews from unlabeled reviews. Second, they employ active learning to label the remaining ambiguous reviews. Third, they use the resulting labeled reviews and the remaining unlabeled reviews to train a transductive SVM classifier. Unlike our method, this study assumes manual intervention in the active learning step.

Wan (2009) employed co-training (Blum and Mitchell, 1998) to exploit labled reviews for a resource-rich language (English) in a document-level sentiment classification task in Chinese. Co-training assumes two independent classifiers that solve the target task, and use the output of one classifier to train the other classifier. With the help of machine translation, they could train two document-level sentiment classifiers in Chinese and English alternately. Their method can be applied to only resource-scarce languages that have a machine translation system between the target resource-scarce languages and a resource-rich language (e.g, English) and the classification performance could be heavily affected by the translation quality.

Yanyan Zhao and Liu (2010) adopted a graph-based propagation approach called Potts model (Wu, 1982) to solve a sentence-level sentiment classification task. Similar to label propagation we used in this study, Potts model uses the relationship among instances, and each instance arrives a probability state through the process of propagation until the whole graph stabilizes. We should mention that the motivation of their study is not to obtain high classification performance in a minimally-supervised setting but to make use of intra- and inter-document evidences in sentence-level sentiment classification. The usefulness of a graph-based semi-supervised algorithm in a minimally-supervised setting remains to be investigated.

3 Proposed Method

3.1 Method Overview

We adopt label propagation (Zhu and Ghahramani, 2002), which is a graph-based semi-supervised learning method, to solve a document-level sentiment classification task. Label propagation owns a

[1] hereafter referred to as sentiment features

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
房间很小，很冷，不满意！	很小 很冷 不满意
The room is very small and cold. Unsatisfying!	Very small, very cold, Unsatisfying
服务态度不错，吃的很好。	态度不错 很好
Service attitude is OK. The food is very good.	Attitude is OK, very good

lot of advantages including convergence and a well defined objective function. It has been successfully employed in several NLP tasks, such as sentiment lexicon induction (Rao and Ravichandran, 2009) and word sense disambiguation (Niu *et al.*, 2005). To the best of our knowledge, there is no previous work that uses this algorithm in document-level sentiment classification where there are only a few amount of training data available.

Our method is divided into three steps:

Step 1: We extract from each review *sentiment features*, which are words/phrases with sentiment polarity, and then represent the review with a vector of extracted sentiment features (*sentiment feature vectors*).

Step 2: We construct a similarity graph by regarding the reviews (sentiment feature vectors) as vertices. The edge (weight) between two vertices (reviews) represents a degree of similarity between their sentiment polarity.

Step 3: Having a few vertices labeled as seeds, each vertex iteratively propagates its label to its neighboring vertices according to their similarity; seeds (initially labeled vertices) thereby behave like sources that push out labels to unlabeled vertices.

In the following sections, we explain each step in detail.

3.2 Step1: Extract Sentiment Feature

Feature selection plays an important role in label propagation, since the similarity score (label consistency) is directly affected by the design of feature vector. We therefore try the following three different ways to obtain the feature vector. Prior to extraction, Stanford Chinese Word Segmenter[2] and Stanford Log-linear Part-Of-Speech Tagger[3] are used to pre-process each review. In what follows, AD, VV, VA, JJ and NN refer to adverb, verb, predicative adjective, adjective and noun, respectively.

content words: We extract content words with the exception of words with pronoun (PN), measure word (M), cardinal number (CD), proper noun (NR), which do not convey sentiment.

phrases: We extract phrases that are likely to express sentiment by using manually-tailored POS patterns. Table 1 lists five POS patterns that we used to extract phrases, along with corresponding phrases extracted by them. These POS patters motivates from an intuition that

[2] http://nlp.stanford.edu/software/segmenter.shtml
[3] http://nlp.stanford.edu/software/tagger.shtml

Table 3: Similarity measure methods

Name	Computing formula
Dice	$\frac{2\|A \cap B\|}{\|A\|+\|B\|}$
Jaccard Index	$\frac{\|A \cap B\|}{\|A \cup B\|}$
Overlap	$\frac{\|A \cap B\|}{\|min(A,B)\|}$
Cosine (tf-idf)	$\frac{A \cdot B}{\|A\|\|B\|}$
Cosine (binary)	$\frac{\|A \cap B\|}{\sqrt{(\|A\| \times \|B\|)}}$

they are common indicators to identify sentiment expressed in reviews. Note that negation is tagged as AD. Turney (2002) used similar feature extracting strategy.

adjective: We extract only adjective words with POS tag VA and JJ.

Table 2 lists two examples of sentiment features extracted from reviews in the dataset we used in our experiment. They are positive and negative reviews in hotel domains, respectively.

3.3 Step 2: Build Similarity Graph

We try several similarity measures to define the similarity between the feature vectors extracted from reviews in Step 1. We assume that the more similar the sentiment polarity of two reviews is, the higher the similarity score between them is. Table 3 lists similarity measures we have used, where A and B are two reviews represented as feature vectors.

Then we construct a similarity graph from both labeled reviews (seeds) and unlabeled target reviews to be classified. In this study, when the similarity score between two reviews is above 0, we create an edge between two vertices corresponding to the two reviews in the graph, and the edge weight is computed by the similarity score. We should note that this graph construction procedure is performed in a totally parameter-free fashion. We do not require precious labeled data (as development set) to tune hyper-parameters as other (semi-)supervised methods do.

3.4 Step 3: Learning on Similarity Graph

Having the similarity graph obtained in step 2, label propagation (LP) (Zhu and Ghahramani, 2002) iteratively propagates a label from the labeled instances to their neighbors. LP can assign labels to the vertices that are connected to labeled vertices on the graph. In other words, if the graph is well-connected, we need a few labeled data to provide a label to a large amount of unlabeled data.

The similarity graph G=$\{V, E, W\}$ consists of vertices V, edges E, and an $n \times n$ weighted similarity matrix $W = [w_{ij}]$, where $n = |V|$. The edge weight w_{ij} is calculated by similarity score between review i and review j. When similarity score w_{ij} between x_i and x_j is larger, they are more likely to have the same label.

Mathematically speaking, LP aims to minimize the following objective function (Rao and Ravichandran, 2009), where l is the number of labeled data, u is the number of unlabeled data, and y_i is a binary label that takes 1 if the vertex is positive and 0 if the vertex is negative.

$$\sum_{i,j=1}^{l+u} w_{ij}(f(x_i) - f(x_j))^2 \textit{ subject to } f(x_i) = y_i \textit{ for } i = 1, 2, \ldots, l \tag{1}$$

The solution to this problem satisfies the following stationary conditions:

$$f(x_i) = y_i, i = 1...l \;\; f(x_j) = \frac{\sum_{k=1}^{l+u} w_{jk} f(x_k)}{\sum_{k=1}^{l+u} w_{jk}} \tag{2}$$

Algorithm 1: Label Propagation

Input : $G = \{V, E, W\}$ is the similarity graph built in section 3.1;
w_{ij} is the elements of W

1 $T_{ij} = P\{j- > i\} = \frac{w_{ij}}{\sum w_{kj}}$
2 Initialize label matrix Y by using seed examples
3 **while** *Y not converge* **do**
4 $\quad Y = T * Y$
5 $\quad$ Row-normalize Y
6 $\quad$ Clamp the seed examples in Y to their original values
7 **end while**

Output : label Matrix Y

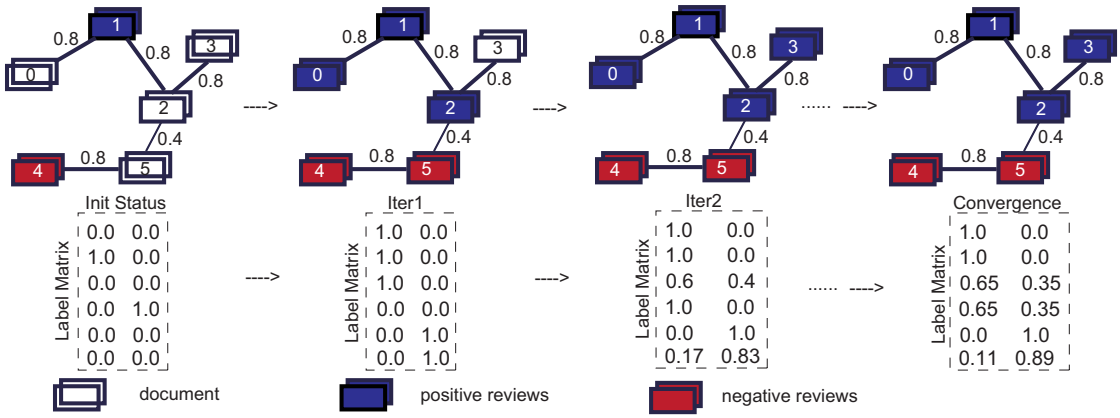


Figure 1: LP illustration

Intuitively, label propagation seeks $f(x_i)$ $(i = l+1, \ldots, l+u)$ that satisfies in the Equation 2 in iterative manner.

Algorithm 1 depicts the label propagation algorithm in detail. There are mainly three steps: it first transforms the given similarity matrix by column normalization (line 1 in Algorithm 1). Each element w_{ij} in similarity matrix W represents the similarity score between review i and review j. After this step we could get a new matrix T whose element T_{ij} represents the transition probability between vertex j and vertex i. Next, the algorithm initializes the label matrix Y, which has N lines and two columns matrix, where N is the number of reviews including both positive and negative ones (line 2 in Algorithm 1). Here we define column 0 and column 1 are used to store a positive value and a negative value respectively. Label matrix Y is also the output of the algorithm. Each line in this matrix is the sentiment polarity score (one positive value and one negative value). The score of labeled positive seed is initialized to (1,0), while negative one is initialized to (0,1). The initialization of the rows of Y corresponding to unlabeled data points is not important (Zhu and Ghahramani, 2002). We initialize the value to (0.5,0.5) which stands for the unlabeled reviews have equal probability to below to positive or negative class. Then, LP propagates labels through the graph, in essence it is an iterative matrix computation. At each interaction, the label matrix Y is row normalized so that each line could be maintained as probability value which stands for the probability that the corresponding review belongs to one specific class. At the end of each iteration, the seeds are re-adjusted to original value, so that we could take seeds as energy sources.

When label matrix Y converges, the propagation process ends and unlabeled target reviews receive both positive and negative polarity value, and both of them are a probability value (the sum is 1). They are categorized according to the two values: for each review, if the value of column 0 (positive polarity value) is larger than column 1 (negative polarity value), the review is classified as positive and vice versa.

Table 4: Classification performance with label propagation

Domain	Sentiment features	Dice	Jaccard	Overlap	Cosine (tf-idf)	Cosine (binary)
Notebook	Adjective words	0.514	0.520	0.507	0.590	0.509
	Content words	0.584	0.681	0.690	0.685	0.611
	Phrases	0.820	0.819	0.819	**0.826**	0.821
Hotel	Adjective words	0.501	0.501	0.501	0.507	0.503
	Content words	0.544	0.534	0.502	0.528	0.532
	Phrases	0.712	0.759	0.735	0.752	**0.764**
Book	Adjective words	0.501	0.501	0.501	0.507	0.503
	Content words	0.626	0.604	0.535	0.626	0.619
	Phrases	**0.641**	0.623	0.627	0.598	0.631

The process of the algorithm is exemplified in Figure 1[4]. Each rectangle stands for one vertex (review). The weight value is the similarity score computed by using similarity measures introduced in Table 3 in Step 1. Here we define blue vertices as positive reviews and red ones as negative reviews.

Algorithms 1 process this graph as follows: Firstly, vertex 1 and vertex 4 are initialized as positive seed and negative seed, respectively. The corresponding label matrix Y is also shown. Then after the the first iteration (Iter1), label from vertex 1 is propagated to vertex 0 and vertex 2. Similar behavior happens between vertex 4 and vertex 5. After the second iteration (Iter2), positive label is also propagated to vertex 3 from vertex 2. Although vertex 2 could also receive negative label from vertex 5, in the corresponding label matrix Y, we could find positive value ($Y[2,0]$) is larger than negative value ($Y[2,1]$). Vertex 2 is still be categorized as positive review according to the classifying criteria we mentioned before. Finally the state of label matrix Y is stabilized (Convergence). The classifying process ends.

4 Evaluation and Discussion

We firstly investigate the performance of sentiment classification using different sentiment features and various similarity measures to define edges in the LP's similarity graph. We then compare LP with representative supervised and semi-supervised algorithms in terms of classification performance when we change the number of labeled reviews.

4.1 Data Set

The dataset we used is ChnSentiCorp[5]. It consists of reviews from three different domains: notebook, hotel, and book. The sentiment polarity of each review has been manually labeled. In each domain, we randomly selected 300 reviews as test dataset. The ratio between positive and negative reviews in the test data is 1:1, so that random baseline achieves accuracy of 0.5. We divided the remaining reviews into two parts: labeled seeds and unlabeled data. The number of labeled seeds is varied from 10 to 300 to control the amount of supervision.

In order to decrease the effect of seed selection, we conduct a ten-rounds experiment for the same number of labeled seeds. In each round, different labeled seeds are used. We report the average classification accuracy as a final result.

4.2 Sentiment Feature and Similarity Graph Selection

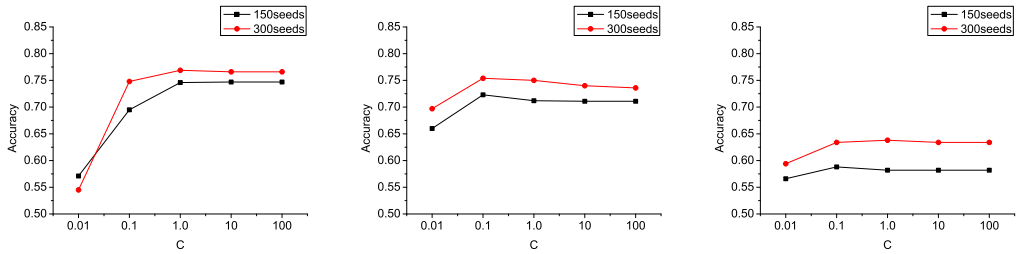
In this section we present classification results of our method when using different sentiment features and similarity graphs. The number of labeled seeds is fixed to 300. The classification results are summarized in Table 4. We have marked the best results using bold characteristics.

[4] Due to the space limitation, we preclude the similarity matrix transformation.

[5] http://www.searchforum.org.cn/tansongbo/corpus-senti.htm

Table 5: Similarity graphs statistics

Domain	Number of vertices	Number of edges	Density	LCS size
Notebook	3632	189,271	0.03	3433
Hotel	3544	346,127	0.06	3372
Book	3631	266,265	0.04	3495



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

Figure 2: C parameter sensitivity in SVM

We observe that when we use phrases as sentiment features, we obtained the best classification performance. On the other hand, when we use adjectives or content words as sentiment features, the classification accuracy drops significantly. We guess that the polarity of adjective can be affected by their contexts (which noun they modify or which adverb they are modified by), while the POS patterns could capture such contexts. Some of the content words (nouns) convey no polarity into the sentence, and they wrongly connect reviews that are in the opposite polarity but share the topics (nouns).

We could find that, in the same domain, the classification performance is not greatly affected by the choice of similarity measures. In the following experiments, we use cosine (binary) as similarity measure.

After determining review representation and similarity measure, we build the similarity graph for reviews in each domain. The statistics of similarity each graph are presented in Table 5.

We could see in all the three domains the largest connected subgraph (LCS) contains most of the vertices, which means the similarity graphs we built are well-connected.

4.3 Performance Comparison

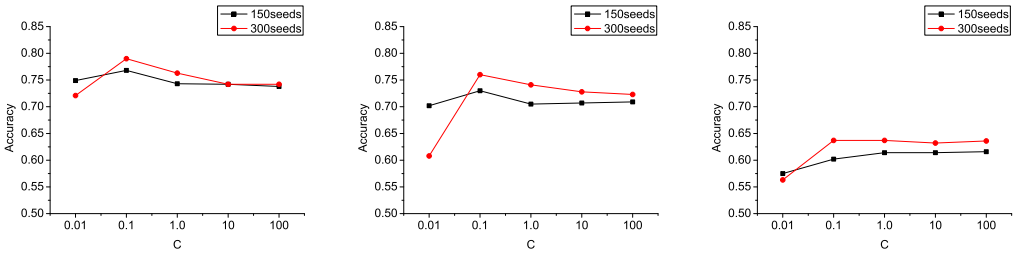
We compare our method with the following two baseline methods:

Support Vector Machines (SVM) (Vapnik, 1995) is a well-known supervised learning algorithm; it tries to construct a hyperplane that classifies positive and negative training examples and then use it for classification. A good separation is achieved by the hyperplane that has the largest distance to the nearest training data points of any class. It has a hyper-parameter C which controls the trade-off between training error and margin. SVM has been already used in a document-level sentiment classification task(Mullen and Collier, 2004).

Transductive Support Vector Machines (TSVM) (Joachims, 1999) is a semi-supervised extension of traditional SVM so that it can exploit unlabeled data in semi-supervised learning. In addition to SVM hyper-parameter C, TSVM has additional hyper-parameter p, which specifies the ratio of positive/negative examples in unlabeled data.

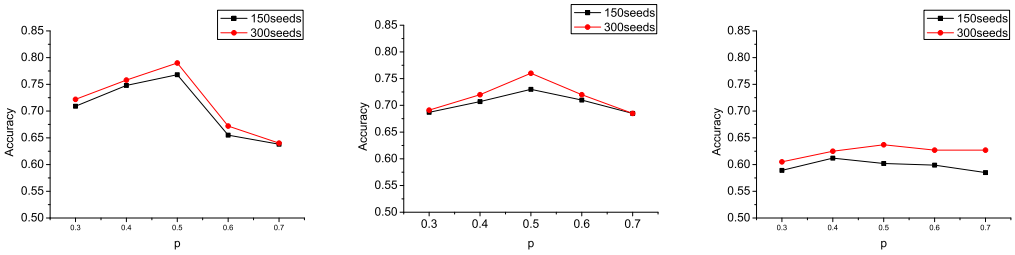
We used SVMlight[6] as implementations of SVM and TSVM used in our experiments.

[6] http://svmlight.joachims.org/



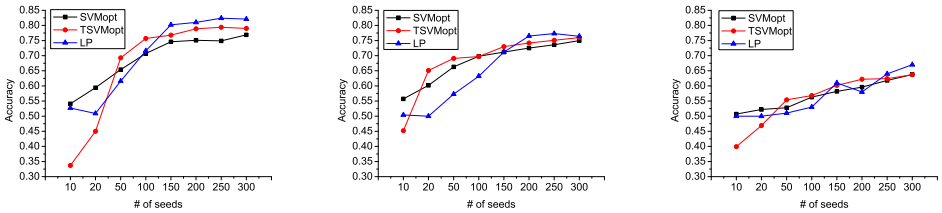
(a) Notebook reviews (b) Hotel reviews (c) Book reviews

Figure 3: C parameter sensitivity in TSVM (p is fixed to 0.5)



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

Figure 4: p parameter sensitivity in TSVM (c is fixed to 0.1)



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

Figure 5: Performance Comparison between LP and optimized SVM ($c = 1.0$)/TSVM ($c = 0.1, p = 0.5$) (SVMopt/TSVMopt)

4.3.1 Parameter Sensitivity We first investigate the effect of the hyper-parameters of SVM and TSVM. The effect of parameter C in SVM and TSVM are shown in Figure 2 and Figure 3 respectively. The vertical axis indicates the accuracy, while the horizontal axis indicates the value of parameter C. The effect of parameter p in TSVM is presented in Figure 4. The vertical axis indicates the accuracy, while the horizontal axis indicates value of parameter p.

We could find the performance of SVM and TSVM are sensitive to the changes of parameter C; they perform worse when the value of C is small. Also, the optimal C value is different in each domain. The performance of TSVM is further affected by hyper-parameter p. It behaves best in the most of situations with p equals to 0.5. This is probably because the unlabeled data we use during the experiment is nearly balanced in positive and negative sentiment. While in the situation where we have no prior knowledge in characteristics of unlabeled data, it is hard to tune hyper parameter p. When we make performance comparison, the value of parameter C in SVM is set to 1.0; the value of C and p in TSVM are set to 0.1 and 0.5, respectively. We found the they perform best in most of situations with these settings. And we should point out that it is usually quite difficult to tune the parameters without development set, which are precious in a minimally-supervised setting.

4.3.2 Performance Comparison The comparison of classification performance is shown in Figure 5; the vertical axis indicates the accuracy, while the horizontal axis indicates the number of labeled seeds. The performance of LP is comparable with SVM and TSVM with optimal hyper-parameter setting (referred to as SVMopt and TSVMopt int Figure 5) and even outperforms them when an appropriate number of seeds are provided. We should note that label propagation is a parameter-free method. It performed reasonably well across all the three domains without tuning hyper-parameters.

We could find that, with the increase of labeled instances, the performance of all of three methods were improved. For SVM and TSVM, the increase of labeled instances means more labeled data are available, so the classifiers perform better. On the other hand, for label propagation when more seeds take part in the label propagating process, unlabeled vertices could get more reliable sources so that LP could become more confident to decide the label one specific vertex belongs to.

Finally, all the three methods do not perform well in book domain. Because in book reviews people would discuss various aspects including the story, the writing style of author, the figures in the book and even the publisher. Sometimes the sentiment those aspects are not consistent. Turney (2002) reported similar findings on movie reviews.

5 Conclusion and Future Work

In this paper we explore the use of label propagation (LP) for a document-level sentiment classification task in resource-scarce languages. We exhaustively investigate the performance of label propagation in this task while varying the way to build the task similarity graph. We found the similarity graph built on phrases is an excellent choice for label propagation for this task. And based on phrases, conine (binary) could be used to measure the sentimental similarity well.

We compared our method with supervised learning and semi-supervised learning methods on real Chinese reviews classification in three domains. Experimental results demonstrated that label propagation showed a competitive performance against SVM or Transductive SVM with best hyper-parameter settings. Considering the difficulty of tuning hyper-parameters in a resource-scarce setting, the stable performance of parameter-free label propagation is promising.

We plan to further improve the performance of LP in sentiment classification especially when we only have a small number of labeled seeds. We will exploit the idea of restricting the label propagating steps when the available labeled data is quite small. Besides, according the work (Velikovich *et al.*, 2010) label propagation suffers a problem when there are dense subgraphs in the similarity graph. Because in label propagation a vertex with multiple paths to a seed will be influenced by all these path, when the label information comes into a dense subgraphs a reinforcement effect will happen. We plan to adopt the idea of damping factor used in PageRank (Brin and Page, 1998) to solve the problem. Finally, we will also explore sentiment feature extracting approaches and different similarity graphs combining methods.

References

Blum, Avrim and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, COLT' 98, pp. 92–100. ACM.

Brin, Sergey and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7), 107–117, April.

Dasgupta, Sajib and Vincent Ng. 2009. Mine the easy, classify the hard: a semi-supervised approach to automatic sentiment classification. In *Proceedings of the Joint Conference of*

the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2, ACL '09, pp. 701–709.

Gamon, Michael. 2005. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *In COLING*, pp. 841–847.

Joachims, Thorsten. 1999. Transductive inference for text classification using support vector machines. pp. 200–209. Morgan Kaufmann.

Manning, Christopher D. and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. MIT Press.

Matsumoto, Shotaro, Hiroya Takamura, and Manabu Okumura. 2005. Sentiment classification using word sub-sequences and dependency sub-trees. In *PAKDD'05*, pp. 301–311.

Mullen, Tony and Nigel Collier. 2004. Sentiment analysis using support vector machines with diverse information sources. In *In Proceedings of Conference on Empirical Methods in Natural Language Processing*.

Niu, Zheng-Yu, Dong-Hong Ji, and Chew Lim Tan. 2005. Word sense disambiguation using label propagation based semi-supervised learning. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pp. 395–402.

Pang, Bo and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2, 1–135, January.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pp. 79–86. Association for Computational Linguistics.

Rao, Delip and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pp. 675–682.

Turney, Peter D. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pp. 417–424.

Vapnik, Vladimir N. 1995. *The nature of statistical learning theory*. Springer-Verlag New York, Inc.

Velikovich, Leonid, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 2010. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pp. 777–785.

Wan, Xiaojun. 2009. Co-training for cross-lingual sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pp. 235–243.

Wu, F. Y. 1982. The potts model. *Rev. Mod. Phys.*, 54(1), 235–268, Jan.

Yanyan Zhao, Bing Qin and Ting Liu. 2010. Integrating intra- and inter-document evidences for improving sentence sentiment classification. *ACTA AUTOMATICA SINICA*, 36(10), 1417–1425.

Zhu, Xiaojin and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical report.