博士論文

Minimally Supervised Approaches to Emotion Classification using Unlabeled Data

（ラベルなしデータを用いた情緒分類への弱教師ありアプローチ）

University of Tokyo

Graduate School of Information Science and Technology

Department of Information and Communication Engineering

48-107411 任勇（Yong Ren）

Supervisor Prof. Masaru Kitsuregawa

13th June 2014

Abstract

Nowadays, emotion classification has attracted much attention due to many interesting applications such as stock market prediction with social media sentiment, emotion aware Text-to-Speech (ToS) system, learn the crowd concerns and opinion and so on.

Conventionally, people address emotion classification in the framework of supervised learning. However, supervised approaches heavily rely on a large amount of linguistic resources, which are costly to obtain for under-resourced scenarios. For example, when we want to analyze the emotion in minor languages or hope to make poll on the sudden and hot events, we cannot put the supervised methods into usage due to the limited training data. So in reality, a minimally-supervised approach is necessary. To the best of my knowledge, rare study has focused on the task of emotion classification in the minimally-supervised setting.

In the first study, I exploit the usage of label propagation (LP), which is graph-based semi-supervised learning (SSL), to overcome the scarce resource problem. Though LP has a lot of advantages such as the convergence, the essentially support for multi-(label) classification, and the high scalability. It is still unclear how to deploy such kind of approach in emotion classification task.

I have evaluated several fundamental and critical problems regarding to the application of LP including the review representation, the similarity measure and compare it with the best tuned baseline systems: SVM and TSVM. The evaluation is done on real Chinese reviews from three domains: notebook, book and hotel. I conclude that phrases extracted by manually designed part-of-speech (POS) patterns is the optimal choice to represent reviews since they can capture the related context well, and the performance of LP is comparable to the performance of best tuned baselines.

Then I also study the other important issues such as controlling label propagation, choosing the initial seeds, selecting edges in the application of graph based SSL. Especially, I propose the effective metrics including PageRank value of vertex and the number of degree to choose seeds. The evaluation on three real datasets demonstrates that manipulating the label propagating behavior appropriately and choosing labeled seeds play a critical role in adopting graph-based SSL approaches for this task. I unveils that both PageRank and the vertex degree can be used to select the seeds. The explanation is that they have close relationship to the dense areas in the whole graph, and it is the dense area that facilitates the classification in graph-based SSL algorithms. Finally, I also confirm that edge pruning is not a reliable way to improve the performance due to the triggering changes on the dense areas.

Be different from the first study, I adjust the focus to the task of feeling prediction on reader viewpoint in the second study. Besides, I investigate the usage of conventionally adopted classifier (SVM) and commonly used features (bag-of-words) such that we do not need to change the existing classification paradigm to suite the minimally-supervised setting. The focus is to make dense feature space via unsupervised feature learning from a large scale unlabeled corpus.

Specifically, I explore the identification of horror episodes, which is helpful to avoid triggering bad experience to users. I formulate horror recognition as a binary classification problem. Especially, I address it in the minimally supervised scenario where only a handful of labeled samples are available. Besides adopting conventional text features including unigrams and bigrams, I attempt to exploit features derived from a large amount of unlabeled corpus in this task. I investigate the usefulness of the induced features through a series of experiments, using newly emerging literature, hint fiction, as typical, concise example of horror episodes.

I begin with the performance comparison on three unsupervised feature learning: brown-clustering which is conventional word clustering algorithm and is able to capture the syntax; LDA which is a well-known approach to extract the topic distribution in the given document, and word embedding which is state-of-the-art feature learning method to encode semantics in word level. I have confirmed the advantage of word embedding over a series of experiments owing to its capability to capture word semantics .

Meanwhile, I have also investigated several factors which have been ignored in NLP research field such as the impact of the size of unlabeled corpus, the impact of methods to utilize the derived features. I have observed that people can benefit from the increase of unlabeled corpus, and I conclude that in the minimal supervised setting, replacing the conventional bag-of-word features with induced dense features is a better way while combination of induced features with BOW features should be preferred when there is a certain number of labeled data. In terms of word embedding , I have confirmed that it would be better to avoid the usage of low dimension word embedding features.

Though word embedding has a lot of advantages, the ignorance of task specific information is its main demerit. In this study, I propose one novel way to integrate supervision into the existing word embedding. The adjustment is conducted through a weighted linear addition in word embedding vector space, just before the formation of dense feature for one document. I have evaluated the effectiveness of my proposal through the empirical study using both customary and automatic constructed supervised vectors. The conclusion is that the word embedding can benefit from task specific information.

In a summary, I focus the application of semi-supervised strategies in the minimally-supervised setting. I propose an effective seed selection for graph-based SSL to stabilize the performance and I propose a novel approach to strengthen the existing word embedding vectors. The effectiveness of both proposals have been evaluated on real Chinese text.

Acknowledgements

Foremost I want to thank my advisor, Prof. Masaru Kitsuregawa. It is my honor to be his PhD candidate for the passed four and a half years. He has taught me, both consciously and unconsciously, how excellent and influential research should be and how meaningful life is. Meanwhile, he also pointed out the research direction for me when I was a low year PhD student. I really appreciate that he is willing to provide an amazing platform for me to explore state-of-the-art technologies. His valuable instructions on basic knowledge and research attitude have been stimulating me.

Special thanks to Dr. Nobuhiro Kaji and Dr. Naoki Yoshinaga. They help me a lot on locating the specific research topics, conducting the experiment, making good representation and writing academic papers. I have learned a lot from their high demand and strictness on research. They set good examples for me as excellent researchers.

I like to express grateful thank to Prof. Masashi Toyoda for his kind advice, helpful assistance on the direction and many issues regarding to my research. Also, sincere thanks to my Ph.D. committee professors, Prof. Shuichi Sakai, Prof. Jun Adachi, and Prof. Kiyoharu Aizawa.

I would also like to thank other members in Kitsuregawa & Toyoda lab, including Dr. Daisaku Yokoyama, Dr. Naohiro Ito, Dr. Zhenglu Yang and Dr. Haichuan Shang, to name few. Thanks for their valuable time allocated for my presentations and they usually provide a lot of useful advices and comments on my study. I also want to thank Mrs. Yoko Izaki and all the other secretaries, for their kind help to complete many documents and affairs. I pass through the challenging and interesting research time with the joy of being in company with, Dr. Yanhui Gu, Mr. Wenliang Gao, Mr. Xingtian Shi and Mr. Hongguang Zheng.

I would express thanks to Dr. Bo Zheng, Dr. Xiangqi Huang, and Dr. Rencheng Zheng. I have learnd a lot including technical knowledge, research approach and experience and so forth from them.

This thesis would not be possible without generous financial support from Japanese Government Scholarship Program (Monbukagakusho Scholarship by JASSO). JASSO also provides student accommodation and other convenient life supports during my stay in Japan.

Finally, I express gratitude for my wife, my family and my parents in law. It is their understanding, perpetual support and unconditional love that make me overcome all the difficulties through the all the years of my study at the University of Tokyo.

To my parents in the heaven!

August 2014

Contents

List of Figures

List of Tables

Abbreviations

Bag of Words	BOW
Consumer Generated Media	CGM
Label Propagation	LP
Latent Dirichlet Allocation	LDA
Modified Adsorption	MAD
Neural Network Language Model	NNLM
Part-of-Speech	POS
Point-wise Mutual Information	PMI
Semi-supervised Learning	SSL
Semantic Orientation	SO
Support Vector Machines	SVM
Transductive Support Vector Machines	TSVM
Text-to-Speech	ToS

Chapter 1

Introduction

1.1 Research Background

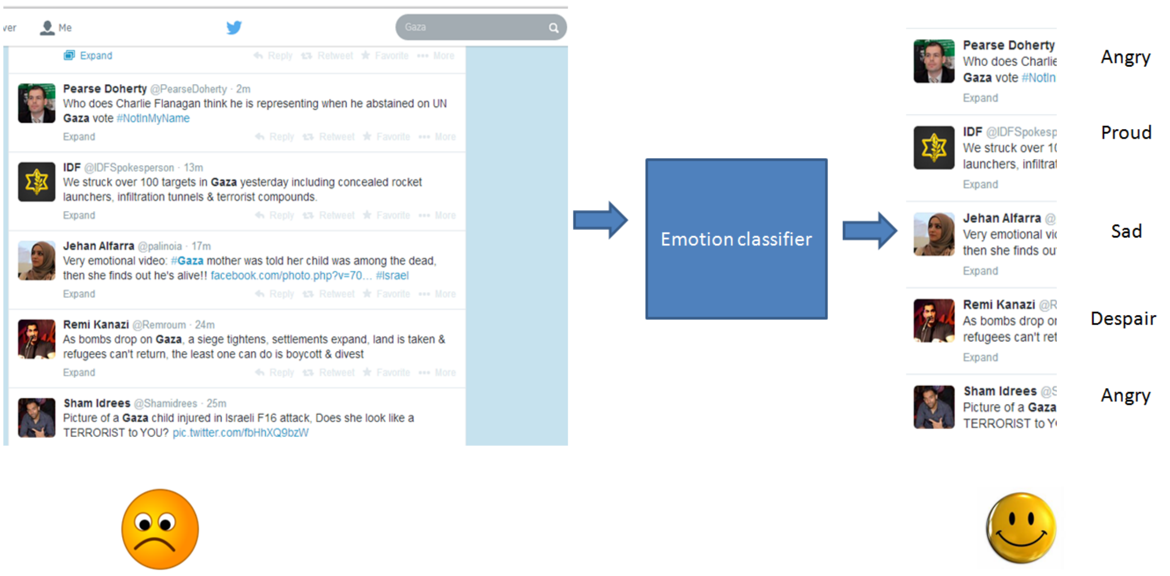
As shown in figure 1.1, document-level emotion classification is the task to classify the documents according to the feeling expressed in the text. Over the last decade, it has attracted much attention from NLP researchers [PL08, PLV02, MC04, MTO05, Gam05, LYC07, LC08, YCS09, HKYT13] since it can play critical role in analyzing the highly developing consumer generated media (CGM).

There are a lot of interesting applications on document-level emotion classification. The foremost application is to pool crowds attitude on specific issues as people are willing to share their thought on CGM platforms nowadays. Figure 1.1 presents several examples where CGM users are declaring their attitudes on very recent events. I also create a demo [2] to analyze the emotion of users on Chinese CGM called Sina weibo [3](a short description is given in Appendix B).

[1]http://www.sntmnt.com/products/

[2]https://github.com/renyong/demo

[3]http://www.weibo.com/



Pearse Doherty @PearseDoherty · 2m
Who does Charlie Flanagan think he is representing when he abstained on UN **Gaza** vote #NotInMyName

IDF @IDFSpokesperson · 13m
We struck over 100 targets in **Gaza** yesterday including concealed rocket launchers, infiltration tunnels & terrorist compounds.

Jehan Alfarra @palinoia · 17m
Very emotional video: #**Gaza** mother was told her child was among the dead, then she finds out he's alive!! facebook.com/photo.php?v=70... #Israel

Remi Kanazi @Remroum · 24m
As bombs drop on **Gaza**, a siege tightens, settlements expand, land is taken & refugees can't return, the least one can do is boycott & divest

Sham Idrees @Shamidrees · 25m
Picture of a **Gaza** child injured in Israeli F16 attack, Does she look like a TERRORIST to YOU? pic.twitter.com/fbHhXQ9bzW

Figure 1.1: Task illustration

The second type of applications locates in market prediction. For example, a corporation called "SNTMNT" [4] claims that they could make a forecast on the stock mark using social media users mood. Figure 1.2 is a screenshot of their product demo, where the yellow line represents social media emotion earlier and the yellow line indicates the stock market value. We can clearly see that the emotion curves reflects the trends market curve very well. Similarly, an Austrian real estate company reports that the trading volume of houses has close relationship with customers's feeling, and the founding is evidenced in figure 1.3. Besides, conventional services can also benefit from document-level emotion classification. German Research Centre for Artificial Intelligence(DFKI) has made a emotion-sensitive text-to-speech (ToS) system named "OpenMary" [5] which can augment the existing ToS service with expressive capability.

[4]http://www.sntmnt.com/
[5]http://mary.dfki.de/

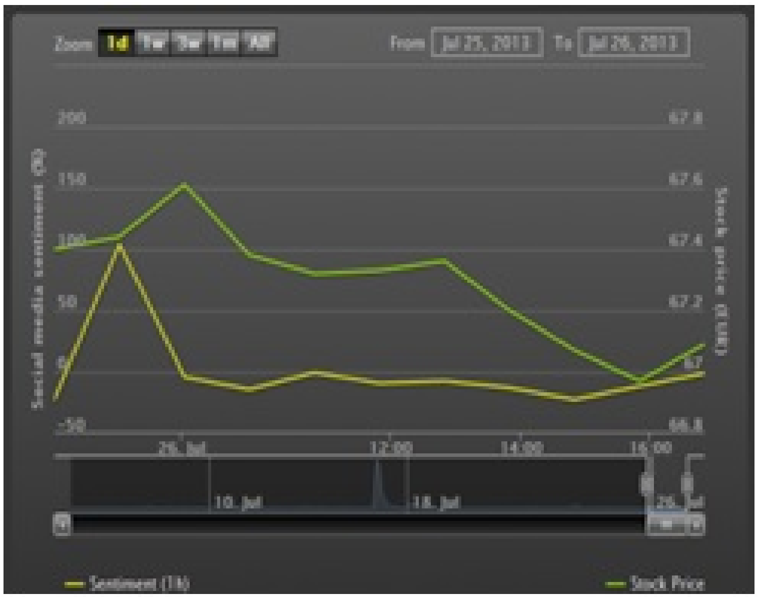


Figure 1.2: Stock market prediction using CGM emotion(screenshot from[1])

Though many studies (surveyed in [Liu12]) have been spawned by those applications, most of the existing methods formulate emotion classification as a supervised classification task and train a reliable classifier from manually labeled data. The main demerit of those supervised approaches is that they demand a large amount of training data to achieve high accuracy.

Unfortunately, for real world scenarios such as text analysis in minor languages or new domains and identification of special feeling in documents, a sufficient amount of training data is not available. The annotation is known to be time consuming and requires substantial human labor by domain experts. Emotion classification is therefore a quite challenging problem for such minimally-supervised setting. The purpose of my study is to explore the strategies for minimally-supervised emotion classification.

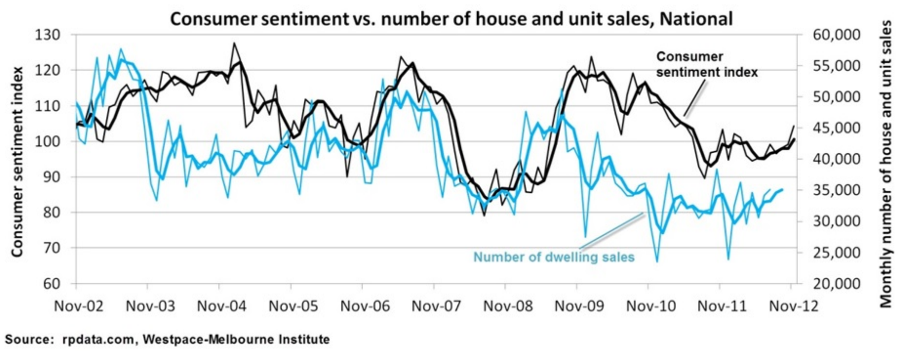


Figure 1.3: Real estate market prediction using CGM emotion

1.2 Problem Definition

In this thesis, I specify emotion classification as to the task of classifying documents according to the overall feeling expressed by the authors or triggered to readers. In other words, both author-centered (section 3) and reader-oriented (section 4) text analysis have been taken into consideration in my study.

Mathematically, the task is to find the discrete functional mapping $F(X, O) \to Y$, where $X = x_1, x_2, ..., x_n$ is a high dimensional vector transformed from original text, $Y = 0, 1, ..., m$ is a discrete value set to represent emotional labels, and $O = o_1, o_2, ..., o_k$ is the optional auxiliary vector to represent outside resources. The goal is to approximate the function F.

1.3 Contribution

I focus the utilization of unlabeled corpus for the minimally supervised emotion classification. The main contributions of our study are summarized as follows:

- I proposed a novel and automatic seed selection approach for graph-based semi-supervised learning, and the sentiment polarity classification conducted on open dataset of Chinese reviews exhibit that my proposal can significantly stabilize the performance especially when the number of seeds is very small. It will be carefully described in section 3.

- I design a method to integrate supervised information into the existing word embedding vectors. Meanwhile, I provide two ways to gain the supervised information. I launch the evaluations on the task of horrible stories identification using real Chinese short texts, and the results present that my approach can further improve the performance. The detail will be explained in section 4.

1.4 Organization of the Thesis

The remainder of the thesis is structured as follows:

- In Chapter 2, I introduce the related works on document-level emotion classification. Besides author-centered (section 2.1) and reader-oriented (section 2.2) studies, I especially present several studies that focus on emotion representation (section 2.3) since representation method is crucial to the whole task.

- In Chapter 3, I begin with the evaluation of the well-known graph-based semi-supervised learning algorithm called label propagation (LP) in the task of sentiment polarity classification. Then I propose the usage of metrics from social network analysis to locate the candidates for labeling. Meanwhile, I also explore the impact of many factors such as sentiment representation method, pruning edges and control on the label propagation.

- In Chapter 4, At the beginning, I investitive the usefulness of dense representation by using features derived from corpus, and three well-known methods are evaluated. In terms of word embedding, I measure the influence of several factors such as the dimension of induced vector and application method. At the last, I propose a novel way to add task specificity into word embedding.

- I finally, summarize this thesis and present future directions.

Chapter 2

Related Work

I will begin with the introduction on the studies that focus on sentiment polarity classification. In the realm of emotion classification, sentiment polarity classification is the mostly addressed task. Then, I will outline readers feeling prediction, which is growing member in the family of emotion classification. Particularly, I will provide an overview on application of unsupervised emotion feature learning at the end of this section. Unsupervised emotion feature learning is subset of unsupervised feature leaning [LPLN09, BC$^+$08, Le13] which accompanies deep learning [HOT06, SHD14] and rides on the crest of a wave in various research areas (surveyed in [Ben09]).

2.1 Sentiment Polarity Classification

Conventionally, supervised learning algorithms [PL08, PLV02, MC04, YZL09] are used to resolve the task of sentiment polarity classification. But as I have introduced in section 1.1, they cannot be directly applied in minimally-supervised scenario. I will provide more explanation on this

[2]http://www.hlt.utdallas.edu/ vince/talks/acl09-sentiment.pdf

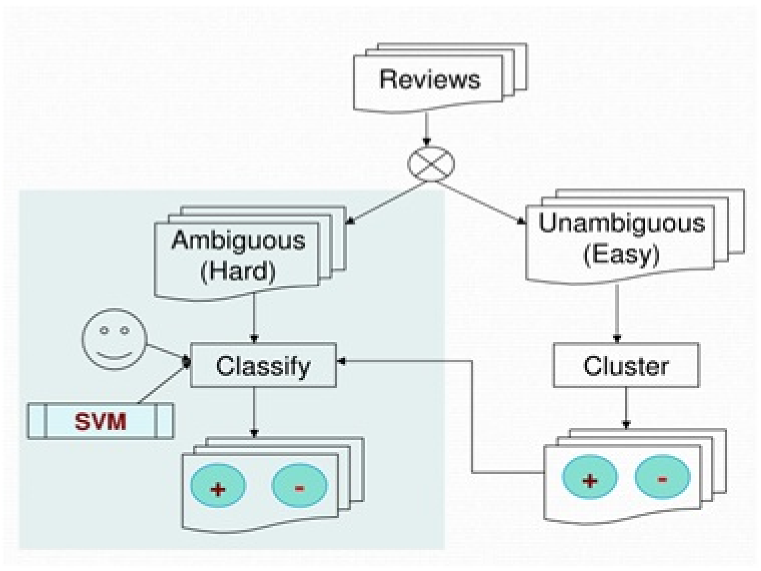


Figure 2.1: Framework of Dasgupta's study (quoted from Page 11 in the author's slide[2])

issue in section 4.2. If the readers want to learn more about the application of supervised learning on sentiment classification, [Liu12] is a comprehensive survey.

In principle, transfer learning [PY10] which is capable to conduct cross-domain learning can be applied in minimally-supervised setting as a resource-scarce domain can benefit from the other domains that are equipped with sufficient training data. Actually, the authors of [PNS+10, BMP06] devise transfer learning approaches to solve cross-domain sentiment classification. However, such kind of attempts still suppose that a large amount of labeled data are available in the other domains, which reduces the application feasibility in reality.

$$PMI(word_1, word_2) = \log_2\left(\frac{P(word_1 \wedge word_2)}{P(word_1)P(word_2)}\right)$$
$$SO(phrase) = PMI(phrase, \text{“}excellent\text{”}) - PMI(phrase, \text{“}poor\text{”}) \quad (2.1)$$

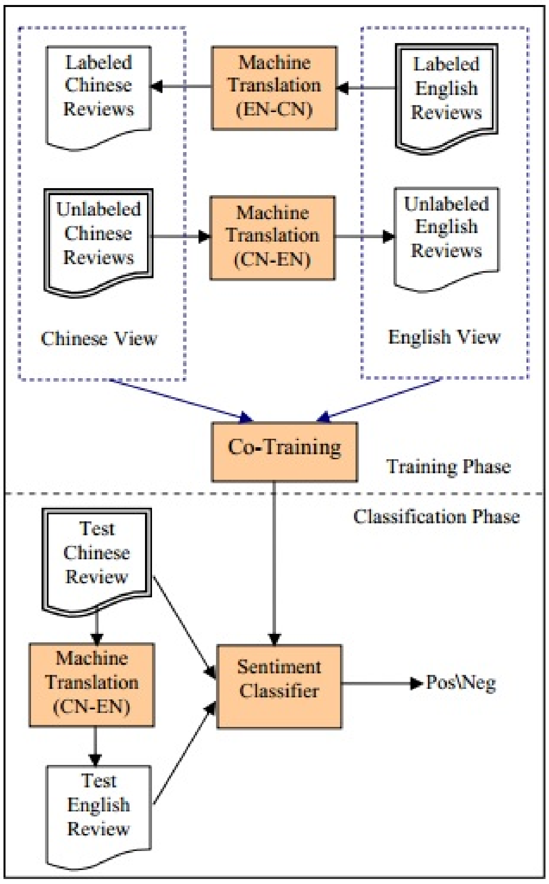


Figure 2.2: Framework of Wan's study (quoted from Figure 1 in [Wan09]))

One extreme campaign for sentiment classification is the application of unsupervised learning paradigm [Tur02a]. At the beginning, the author choose two words "excellent" and "poor" as seeds. Moveover, the phrases extracted using manually-specified part-of-speech (POS) patterns are used as sentiment features. Then, the author calculated the semantic orientation (SO) for each phrase based on point-wise mutual information (PMI). Formula 2.1 presents the computing procedure. Finally, the sentiment polarity of one document is determined by averaging the SO scores among all the phrases in this document. Unsupervised approach never consider the domain specifics,

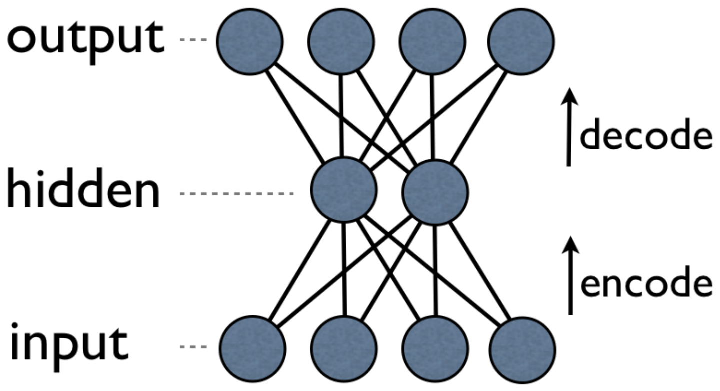


Figure 2.3: Diagrammatic sketch for autoencoder

whereas sentiment usually is domain related [LXWZ13].

Transductive SVM (TSVM) [Joa99], which is a semi-supervised variant of SVM, is used in [DN09] to conduct the document-level sentiment classification task with the similar setting as in my study. As can be seen in figure 2.1, their method is divided into three steps. At first, they performed spectral clustering to identify unambiguous reviews. Secondly, 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 use a SSL-based approach to tackle the scarce resource problem in sentiment classification, they assume manual intervention in the active learning step.

Co-training [BM98] is employed in [Wan09] to exploit labled reviews for a resource-rich language (English) in a document-level sentiment classification task in Chinese. The framework of their study is shown in figure 2.2. Co-training assumes two independent classifiers that can be used to solve the same 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 English and Chinese 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 largely affected by the quality of machine translation service.

Modified adsorption (MAD), which is a graph-based SSL algorithm, is adopted in [SSUB11] 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.[3]

In [SM08], 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 that the proposed method can assign sentiment polarity to both words and documents jointly. Their main focus is to encode prior lexical knowledge into the SSL paradigm with the help of regularized least squares.

A graph-based propagation approach called Potts model [Wu82] [YZL10] adopted to solve a sentence-level sentiment classification task. Similar to label propagation I adopt 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. I should point out 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

[3]http://www.cs.pitt.edu/mpqa/subj_lexicon.html

setting remains to be investigated.

In summary, these studies demand auxiliary resources or substantial human effort. To the best of our knowledge, my study stated in section 3 is the first attempt to apply graph-based SSL algorithms for sentiment classification in a real minimally-supervised setting.

2.2 Readers Feeling Prediction

More recently, prediction on reader reaction has attracted much attention from NLP researchers [LYC07, LC08, YCS09, HKYT13, Bho09]. Usually, the readers' emotion can be classified into several universal types including anger, disgust, fear, happiness, sadness, and surprise [EFP75]. It is generally believed that such kind of study can be helpful to better learn the customers; for example, [WBL$^+$13] has confirmed that the readers' reaction triggered by words can play important role in the click-through rate (CTR) of online advertisement.

[LYC07] initiated the task of emotion classification on readers' standpoints. They exploited Naive Bayes and SVM to categorize news articles according to readers' emotion, where the emotions are *happy, angry, sad, surprised, heartwarming, awesome, bored* and *useful*. They investigated the influence of different feature combinations among Chinese character bigrams, words, metadata of articles and the emotion dictionary, and concluded that the best performance generates from the combination of all those features.

[TC11] explored how the writer emotion affects readers' emotion. Their evaluation was conducted on a microblog dataset. Besides textual features, they also utilized three kinds of non-linguistic features: social relation between the writer and the reader, reader behavior, and relevance between the original post and the comment. They found that those beyond-text features are helpful in predicting the emotion of readers. Similarly, [HKYT13] expanded the research domain to addresser/addressee in online dialogue. They induced additional features from the lister's previous utterance, which is uniquely available in the dialogue dataset.

Different from those previous study using supervised learning approaches, [YCS09] extended Latent Dirichlet Allocation (LDA) [BNJ03] to capture the blogsphere characteristics such as the authorship and reader reaction, and then they make prediction on the response for the political blog passages by using the proposed model. They discovered that the topic model is promising in predicting the reader emotion.

In terms of the study focusing on discerning one specific perception from the others, [MS05] made use of Naive Bayes and SVM to distinguish humorous text from non-humorous text. They only take one sentence joke ("one-liner") into consideration in their study. The excellent result obtained in distinguishing humorous one-linear from new titles or from proverbs. However, when the negative instances are British National Corpus (BNC) which are text in mixed form, the performance highly degenerates, which indicates the difficulty in capturing humor in text.

To summarize, compared with authors's attitude classification, readers's feeling predication is still in early stage. Most of the related studies inherit the supervised learning custom and focus on integration of diverse beyond-text features to resolve the problem. The challenge caused by limited training data is barely addressed in the research filed.

2.3 Application of Feature Learning

[MDP+11] design a probability model that share similar basis with Latent Dirichlet Allocation (LDA) [BNJ03]. The appealing point is that their model mixes both unsupervised and supervised technologies to capture the sentiment similarity between two given words. However, this method requires the corpus of labeled reviews, and there are bottlenecks in in terms of the speed of training model and scalability.

Both [SPH+11] and [GBB11] tried to encode sentiment in the text into a compact way based on autoencoder [Ben09]. While the authors in [SPH+11] focused on extracting the high level and general sentiment representation that could be used in different review domains, the purpose of [GBB11] is to capture the multi-dimensional complexity of sentiment. The diagram in figure 2.3 illustrates the mechanism of autoencoder. In principal, the input and the output are the same, and there are usually several hidden layers. One hidden layer can be considered as a dense representation for the input. Put it simply, the hidden layer encodes the necessary information of the input data, therefore it automatically extract the features from the input data. It worth pointing out that no labeled information is required in the model training process.

Different from the studies with the purpose of constructing model, [LL13] aims at making use of existing word embeddings. Under their named re-embedding framework, they designed a unconstrained optimization method with a convex objective, and the convex objective is one kind of difference between the target and the source embedding.

Similar to my unsupervised feature deriving strategy in my second study (section 4), the study [TRB10] extensively compared several unsupervised feature learning algorithms including Brown clustering [BDM+92], Collobert and Weston embeddings [CW08], and HLBL embeddings [MH08], in the tasks of NER and chunking.

The shortage of text context drive people to induce auxiliary features from a large scale of corpus. Actually, deriving features from unlabeled document collection has attracted increasing interest in the NLP field [TRB10, LW09]. These techniques alleviate the data sparsity issue, and many works have demonstrated excellent results in corresponding tasks [MGZ04, KCC08].

Chapter 3

Graph-based Semi-supervised Learning with Seed Selection

3.1 Introduction

Semi-supervised learning (SSL) algorithms are attractive approaches to address the minimally-supervised challenge. 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 [ZG09]) have attracted a great deal of attention from research communities.

In this section, I focus on document-level sentiment classification under a minimally-supervised setting, where we only have a few labeled reviews given a priori. I explore two representative graph-based SSL algorithms (basic and state-of-the-art), label propagation (LP) [ZG02] and modified adsorption (MAD) [TC09], to understand the behavior of graph-based SSL algorithms in this task setting. I 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.[1] I obtained the following findings through the 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.
- Pruning edges does not achieve a similar level of performance like in the choice of seeds.

3.2 Graph-Based Semi-Supervised Learning

I 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 this study, x_i is represented by a feature vector, while y_i is the sentiment polarity of the review, *i.e.,* positive or negative. I assume there is no neutral category in this study, so in essence, the 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 I will later explore it in section 3.3.2.

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

In this section, I explain two graph-based SSL algorithms, label propagation (LP) [ZG02] and modified adsorption (MAD)[TC09], which I used in our work.

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, I have $n_l + n_u = |V|$. I 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, $v \in V_l$, I 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.

3.2.1 Label Propagation

The first graph-based SSL algorithm I 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 [RR09, NJT05, VBGHM10a].

Mathematically, LP aims at minimizing the following objective function with respect to the labels that each vertex would own [ZG02].

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

$$\begin{aligned} &\frac{1}{2}\sum_{v,v' \in V} \mathbf{W}_{vv'}(\hat{\mathbf{Y}}_v - \hat{\mathbf{Y}}_{v'})^2 \\ &\textit{subject to } \hat{\mathbf{Y}}_v = \mathbf{Y}_v(v \in V_l) \end{aligned} \tag{3.1}$$

Equation 3.1, 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 equation 3.1 satisfies the following stationary conditions.

$$\begin{aligned} \hat{\mathbf{Y}}_v &= \mathbf{Y}_v \ (v \in V_l) \\ \hat{\mathbf{Y}}_v &= \frac{1}{d_v}\sum_{v'} \mathbf{W}_{v'v}\hat{\mathbf{Y}}_{v'} \ (v \in V_u) \\ \textit{where } d_{v'} &= \sum_{v} \mathbf{W}_{vv'} \end{aligned} \tag{3.2}$$

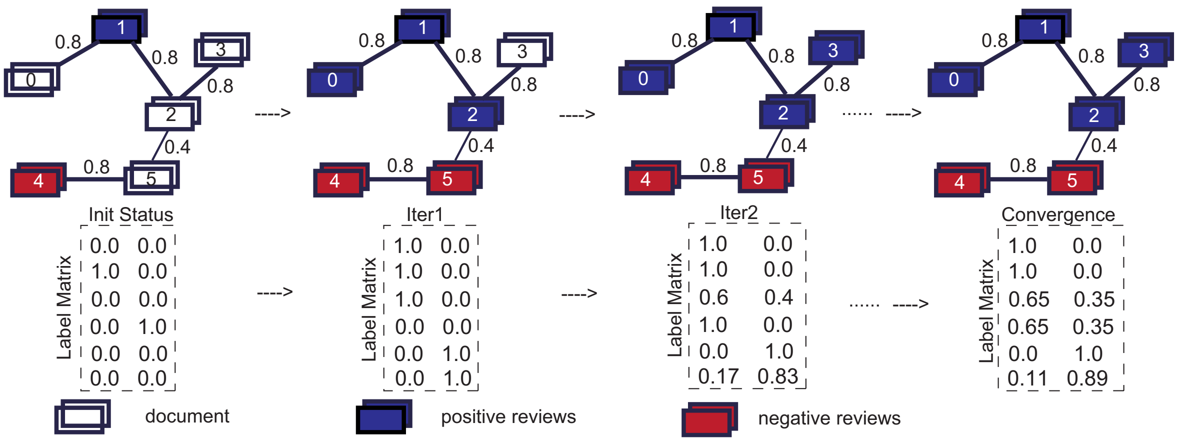


Figure 3.1: LP illustration

Equation 3.2 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 $\mathbf{D} = \text{diag}(d_v)$. Then, I can seek the $\hat{\mathbf{Y}}_v (v \in V_u)$ that satisfies equation 3.2 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 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.

The process of the algorithm is exemplified in Figure 3.1. Each rectangle stands for one vertex (review). The weight value is the similarity score computed by using similarity measures will be introduced in section 3.3.2. Here I define blue vertices as positive reviews and red ones as negative reviews.

Note that LP will suffer from densely connected components in the similarity graph, especially when the weights of edges are not reliable

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     $\mathbf{D}_v = \frac{\sum_u \mathbf{W}_{uv} \hat{\mathbf{Y}}_v}{\sum_u \mathbf{W}_{uv}}$ for $v \in V$
4     for $v \in V$ do
5         $\hat{\mathbf{Y}}_v = p_v^{inj} \times \mathbf{Y}_v + p_v^{cont} \times \mathbf{D}_v + p_v^{abnd} \times \mathbf{r}$
6     end
7 end
  Output: $\hat{\mathbf{Y}}_v$
```

[VBGHM10b]. I guess that high-degree vertices are the origin of that problem and explore the way adsorption tackles the problem in the following subsection.

3.2.2 Adsorption

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

Conventional graph-based SSL algorithms such as LP suffer from *topic drift* caused by high degree vertices [BSS+08]. 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

$$\hat{\mathbf{Y}}_v = p_v^{inj} \times \mathbf{Y}_v + p_v^{cont} \times \sum_{v':(v',v)\in E} \Pr[v'|v]\hat{Y}_{v'} + p_v^{abnd} \times \mathbf{r}$$
$$where \ \Pr[v'|v] = \frac{\mathbf{W}_{v'v}}{\sum_{u:(u,v)\in E} \mathbf{W}_{uv}} \tag{3.3}$$

Algorithm 2 illustrates the adsorption algorithm. I 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.

The values of probability vectors denoted by p_v^{inj}, p_v^{cont}, and p_v^{abnd} play a crucial role. While I manually adjusted those hyper-parameters to investigate the sensitivity, I here introduce an automatic approach proposed in [TC09] 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 [TC09].

The entropy of the transition probability is defined in the formula 3.4.

$$\begin{aligned} H(v) &= -\sum_{u:(u,v)\in E} \Pr[u|v] \log \Pr[u|v] \\ \Pr[u|v] &= \frac{\mathbf{W}_{uv}}{\sum_u \mathbf{W}_{uv}} \end{aligned} \tag{3.4}$$

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

$$f(x) = \frac{log\beta}{\log(\beta + e^x)} \tag{3.5}$$

The function $f(x)$ defined in Eq. 3.5 and is a monotonically decreasing function.

$$g_v = f(H_v) \tag{3.6}$$

$$h_v = (1 - g_v)\sqrt{H_v} \tag{3.7}$$

Obviously, g_v and h_v are respectively proportional and inversely proportional to the entropy defined in Eq. 3.5. By using g_v and h_v, the probability p_v^{cont} and p_v^{inj} is defined as

$$p_v^{cont} = \frac{g_v}{\max(g_v + h_v, 1)} \tag{3.8}$$

$$p_v^{inj} = \frac{h_v}{\max(g_v + h_v, 1)} \tag{3.9}$$

$$p_v^{abnd} = 1 - p_v^{inj} - p_v^{cont} \tag{3.10}$$

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

3.2.3 Modified Adsorption (MAD)

Despite the advantage adsorption owns, as pointed out in [TC09], 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, I will depict the formalization of the final objective function in MAD.[2]

There are three factors that are considered in MAD: the labels predicted and the priori for the seeds should be consistent (Equation 3.11), similar vertices bear the same labels (Equation 3.12), and regularization should be performed (Equation 3.13).

The purpose of equation 3.11 is to keep the consistency between the

[2]Interested readers may refer to the detailed derivation in [TC09].

predicative results ($\hat{\mathbf{Y}}_l$) and the corresponding labeled instances $\mathbf{Y}_l$.

$$\sum_v p_v^{inj} \sum_l (\mathbf{Y}_{vl} - \hat{\mathbf{Y}}_{vl})^2 = \sum_l (\mathbf{Y}_l - \hat{\mathbf{Y}}_l)^{\mathrm{T}} \mathbf{S} (\mathbf{Y}_l - \hat{\mathbf{Y}}_l) \tag{3.11}$$

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).

$$\begin{aligned} \sum_{v,u} \mathbf{W}'_{v\mu} \left|\left|\hat{\mathbf{Y}}_v - \hat{\mathbf{Y}}_u\right|\right|_2^2 &= \sum_l \sum_{v,u} W'_{vu} (\hat{\mathbf{Y}}_{vl} - \hat{\mathbf{Y}}_{ul}) \\ &= \sum_l \hat{\mathbf{Y}}_l^{\mathrm{T}} \mathbf{L} \mathbf{Y}_l \end{aligned} \tag{3.12}$$

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. 3.12 is to distribute labels smoothly across the graph.

Equation 3.13 takes the responsibility of regularization.

$$\sum_{vl} (\hat{\mathbf{Y}}_{vl} - \mathbf{R}_{vl})^2 = \sum_l \left|\left|\hat{\mathbf{Y}}_l - \mathbf{R}_l\right|\right|_2^2 \tag{3.13}$$

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:

$$\begin{aligned} C(\hat{\mathbf{Y}}) &= \sum_l [\mu_1 (\mathbf{Y}_l - \hat{\mathbf{Y}}_l)^T \mathbf{S} (\mathbf{Y}_l - \hat{\mathbf{Y}}_l) \\ &+ \mu_2 \hat{\mathbf{Y}}_l^T \mathbf{L} \mathbf{Y}_l + \mu_3 \left|\left|\hat{\mathbf{Y}}_l - \mathbf{R}_l\right|\right|_2^2] \end{aligned} \tag{3.14}$$

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 [TC09].

3.3 Application

In order to apply graph-based SSL in sentiment classification, my approach is divided into the following three steps:

Step 1: I 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: I 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.

3.3.1 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. I therefore try the following three different ways to obtain the feature

Table 3.1: 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 3.2: Reviews and their sentiment phrases

Reviews	Sentimental features extracted
服务态度不错，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)

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: I 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: I extract phrases that are likely to express sentiment by using

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

Table 3.3: 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)	$\frac{\|A \cap B\|}{\sqrt{}(\|A\| \times \|B\|)}$

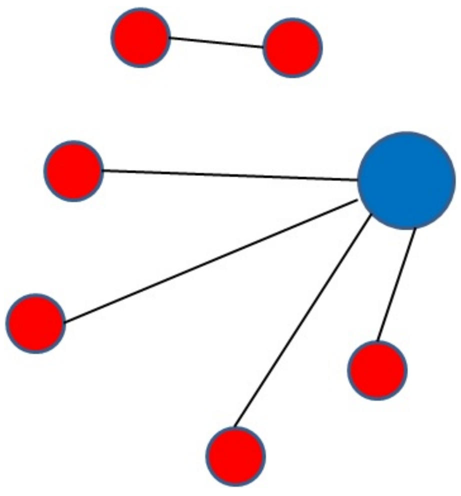
manually-tailored POS patterns. Table 3.1 lists five POS patterns that I used to extract phrases, along with corresponding phrases extracted by them. These POS patters motivates from an intuition that they are common indicators to identify sentiment expressed in reviews. Note that negation is tagged as AD. [Tur02b] used similar feature extracting strategy.

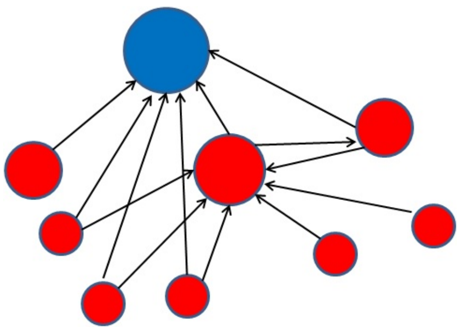
adjective: I extract only adjective words with POS tag VA and JJ.

Table 3.2 lists two examples of sentiment features extracted from reviews in the dataset I used in the experiment. They are positive and negative reviews in hotel domains, respectively.

3.3.2 Step 2: Build Similarity Graph

I try several similarity measures to define the similarity between the feature vectors extracted from reviews in Step 1. I assume that the more similar the sentiment polarity of two reviews is, the higher the similarity score between them is. Table 3.3 lists similarity measures I have used, where A and B are two reviews represented as feature vectors.


(a) Degree


(b) Pagerank

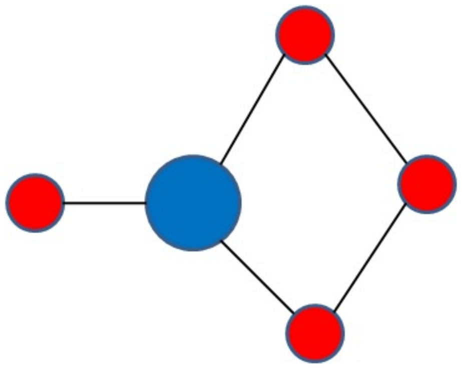

(c) Betweenness centrality

Figure 3.2: Seed selection metrics

Then I 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, I create an edge between two vertices corresponding to the two reviews in the graph, and the edge weight is computed by the similarity score. It worth noting that this graph construction procedure is performed in a totally parameter-free fashion.

3.4 Seed Selection Approach

In a minimally-supervised 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, I explored three criteria for selecting seeds: the number of degrees, the PageRank [PBMW99] value, and the betweenness centrality [Bra01]. Figure 3.2 describes their roles in the graph respectively.

I compared them with the randomly chosen seeds. Note that I also need to balance the sentiment polarity of seeds selected as I did in section 3.5.7; otherwise, unbalanced classification would occur. To meet this requirement, I can scan the vertices (measured by using the number of degrees, the PageRank value and the betweenness centrality, respectively) and annotate them until I accumulate a certain number (10 in our case) of labeled seeds for each class.

3.5 Evaluation and Discussion

I start by introducing datasets followed by an explanation of pre-processing them for evaluation. I 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.

3.5.1 Setting

The datasets I used in the following experiments were from ChnSentiCorp (de-duplicate version).[3] 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, I 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. I 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 I choose a small number reviews to label (here 20, minimum), I ran the experiments ten times by randomly choosing reviews to be labeled. I report the average of the classification accuracy as the final result.

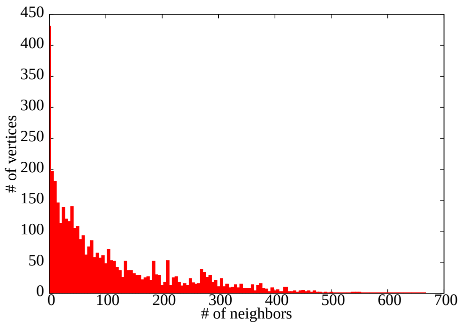
To explore the advantage of graph-based SSL algorithms over supervised counterparts, I used SVM [Vap95], which is a widely-used supervised classification algorithm, as a baseline.

I used SVMlight[4] as the implementation of SVM in our experiments, while I adopted Junto[5] as the implementation of LP and MAD.

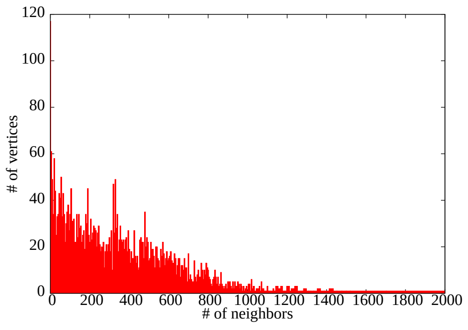
[3]http://www.searchforum.org.cn/tansongbo/corpus-senti.htm

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

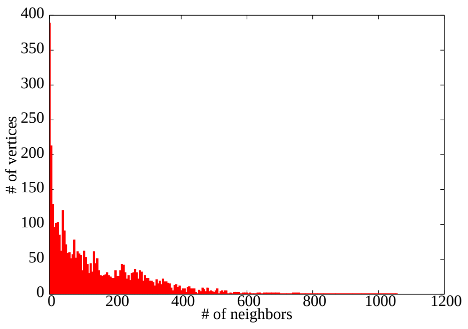
[5]https://github.com/parthatalukdar/junto



(a) Notebook reviews



(b) Hotel reviews



(c) Book reviews

Figure 3.3: Vertex degree distribution

3.5.2 Pre-processing

In this section, I 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. I 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, I follow [RKY+11] to choose phrases with specified POS patterns as sentiment features (prior to extracting the specific phrases, Stan-

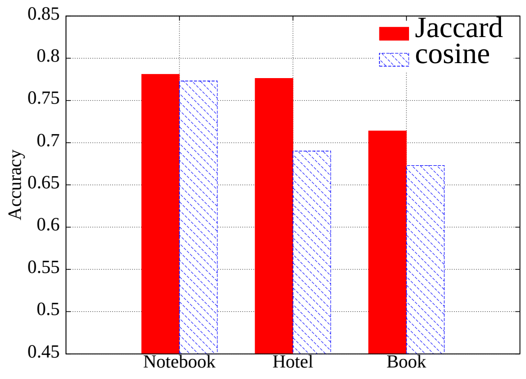


Figure 3.4: Comparison between Jaccard and cosine similarity

ford Word Segmenter[6] and Log-linear Part-Of-Speech Tagger for Chinese[7] are used to pre-process each review).

Table 3.1 lists the five POS patterns [RKY+11] used to extract phrases along with corresponding ones extracted by them, and Table 3.2 lists positive and negative reviews in the hotel domain of the dataset along with corresponding feature representations.

3.5.3 Objectives of Experiments

The followings are the objectives of the evaluations.

- Show the influence of the similarity measure and similarity graph building methods (section 3.5.4).
- Investigate the impact of selecting seeds and pruning edges (section 3.5.5 and section 3.5.6).
- Compare the performance of LP and MAD with SVM in the document-level sentiment classification task when limited training data are avail-

[6]http://nlp.stanford.edu/software/segmenter.shtml
[7]http://nlp.stanford.edu/software/tagger.shtml

Table 3.4: Classification performance with label propagation

Domain	Features	Dice	Jaccard	Overlap	Cosine (tf-idf)	Cosine (binary)
Notebook	Adjectives	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	Adjectives	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	Adjectives	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

able (section 3.5.7).

- Demonstrate the impact hyper-parameters (section 3.5.9).

3.5.4 Sentiment Feature and Similarity Graph Selection

In this section I 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 3.4. I have marked the best results using bold characteristics.

I observe that when I use phrases as sentiment features, I obtained the best classification performance. On the other hand, when I use adjectives or content words as sentiment features, the classification accuracy drops signifi-

Table 3.5: 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

cantly. I 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 (noun-s) 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, I use cosine (binary) as similarity measure.

After determining review representation and similarity measure, I build the similarity graph for reviews in each domain. The statistics of similarity each graph are presented in table 3.5.

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

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, I build an edge between them. A performance comparison (300 labeled seeds) between the two common similarity measures, Jaccard

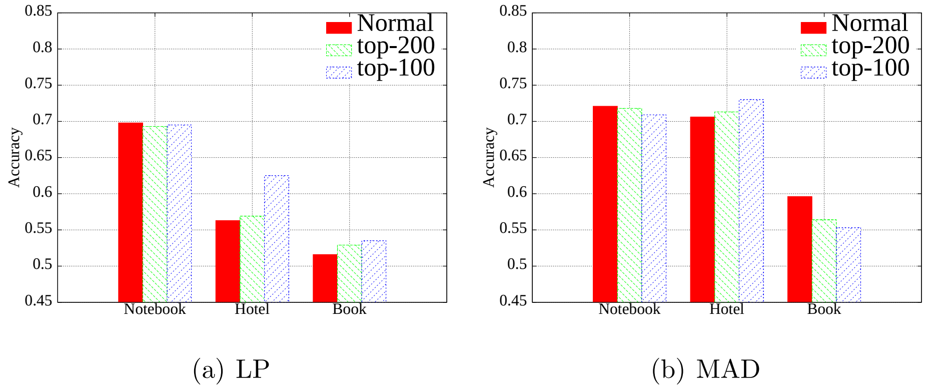


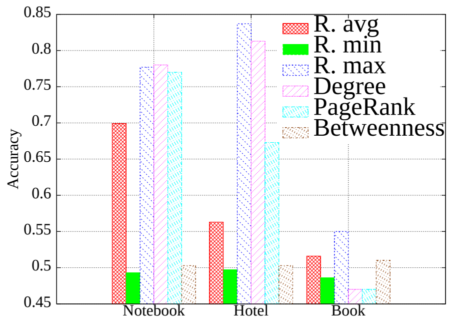
Figure 3.5: Impact of edge selection

similarity coefficient and cosine similarity (with TF-IDF feature weighting), is shown in figure 3.4. I can observe that Jaccard is a better option. Interested readers may refer to [RKY+11] to see a comparison among various similarity measures.

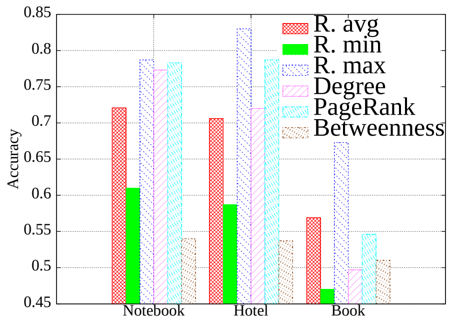
Table 3.5 contains the statistics on the similarity graphs in the three different domains. We can find the largest connected component included all the vertices in the similarity graph, which allowed not only LP but also MAD to label all the reviews. Figure 3.3 shows the degree distribution of the vertices. 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 section 3.5.5), we should take the degree distribution into consideration.

3.5.5 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.



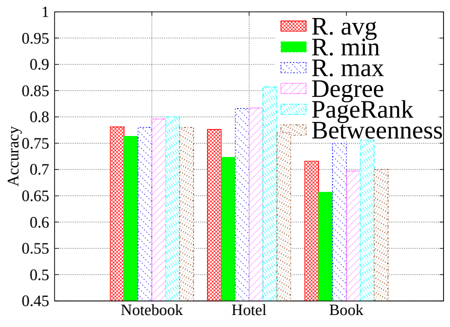
(a) LP



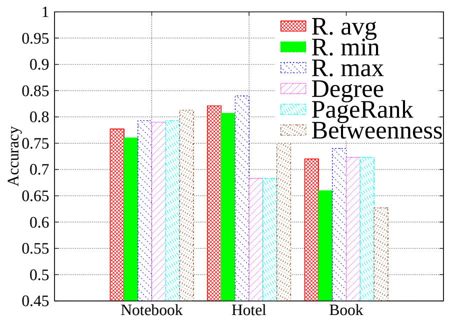
(b) MAD

Figure 3.6: Impact of seed selection (20 seeds)

I thus explore the effectiveness of the method of selecting proper edges. I explored the following strategy to prune unreliable edges. First, given one vertex, I rank its neighboring edges in accordance with the weight, and then, I keep the top-N edges. As I showed in figure 3.3, the degree distribution varied in the different domains. Hence it is obvious that I had more candidates in the hotel domain than in the other two domains. Taking this into account, I left top-100 and top-200 edges for each vertex. The rational is that I 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 I again ran experiments ten times while varying randomly-chosen initial seeds and averaging the obtained accuracy. I present the impact of pruning unreliable edges in figure 3.5. Compared with the case where edge selection was not performed, there was moderate improvement. However, we could not get a clear and consistent tendency. I also conducted a statistical significance test (t-test) on the results, and I observed that all the p-values were above 0.2. I conclude that pruning unreliable edges is not an effective strategy to improve the performance.



(a) LP



(b) MAD

Figure 3.7: Impact of seed selection (300 seeds)

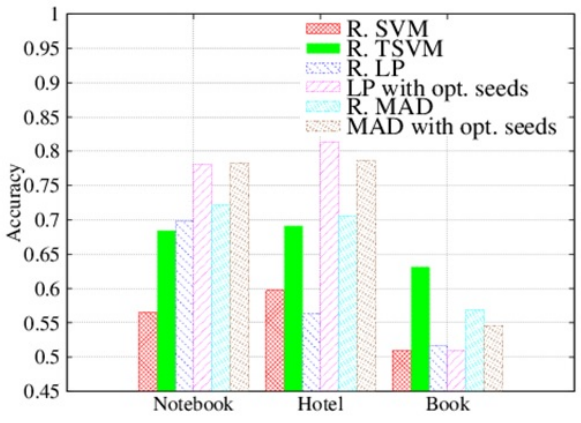


Figure 3.8: Performance improvement (20 seeds)

3.5.6 Impact of Selecting Seeds

Figure 3.6 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. I 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 I merely hope to label a small number of data. I 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.

Finally, though the focus of my study is minimally-supervised setting which means I should limit the number of ladled data as small as possible, I still conduct the seed selection in the case of 300 labeled data so that the readers could have a complete understanding on the effectiveness on the seed selection. Figure 3.7 presents the results where I could clearly see that the performance using seed section is still effective.

3.5.7 Performance Comparison

In this section, I used the Jaccard similarity coefficient [Jac12] to compute the similarity between two reviews. A comparison of classification performance is shown in figure 3.9, where the vertical axis indicates the classification accuracy, and the horizontal axis indicates the number of labeled reviews. Here, I set the hyper-parameters in MAD as the default values. The impact of these hyper-parameters will be shown in section 3.5.9.

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 "misconnection" 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. I 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 I 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 [Tur02b] reported similar findings for movie reviews.

I show the performance deviation in figure 3.10, 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). I explored strategies such as pruning

Table 3.6: Statistics on error types

Error type	False pos	False neg
Type A	19	19
Type B	10	43

unreliable edges (Section 3.5.5) and selecting seeds (Section 3.5.6) to relieve the performance sensitivity.

3.5.8 Error Analysis

I will introduce error analysis in the section. Here, I focus on book domain as the performance is suboptimal among the three domains. I set the number of labeled seed to 300 and adopt PageRank score to locate the initial seeds so that I could uncover the limitedness of graph-based SSL.

Figure 3.11 illustrates two types of errors named as "type A" and "type B" respectively and table 3.6 summarizes the statistics on the errors. Specifically, errors of type A occur when the review is surrounded by many reviews from the oppositive class. In terms of errors of type B, even one reviews is connected to the other reviews from the same class, it is still can be misclassified if the neighboring reviews is classified wrongly. In other words, the error information could be diffused through the graph. Moreover, I have found that errors of type A could trigger errors of type B.

The culprit locates in the fact that the mixed expression in book reviews. A representative example is listed in as follows:

> expect it very much; cheat people

Obviously, the first half of this review is positive appraise on the book whereas the latter part is a complain on the publisher. Such kind of mingled

Table 3.7: Hyper-parameters investigated

Algorithm	Hyper-parameter	Description
MAD	μ_1	weight for keeping original label for labeled data
	μ_3	weight for giving up label diffusion
SVM	C	trade-off between training error and margin
TSVM	C	trade-off between training error and margin
	p	ratio of pos./neg. examples in unlabeled data

Table 3.8: Optimal value of hyper-parameter

Domain	μ_1	μ_3	C in SVM	C in TSVM	p in TSVM
Notebook	1.0	10.0	1.0	0.01	0.5
Hotel	1.0	10.0	0.1	1.00	0.5
Book	10.0	100.0	0.1	1.00	0.5

feeling statement cause the happenings depicted in the figure 3.11. Admittedly, the adopted graph-based SSL algorithms cannot handle the mixed feeling comments. As a matter of fact, the sentiment polarity is even difficult for human to decide. Aspect-based or targeted-oriented studies [KZC+13, BE10] which is beyond the scale of my study could be the potential strategy to resolve this issue.

3.5.9 Hyper-parameter Tuning and Analysis

Here, I reveal the procedure of tuning hyper-parameters in each SSL algorithms and explain the impact of those hyper-parameters on classifying accuracy. Performances of five different sizes of labeled reviews are shown.

I confirm the believe that the optimal values of hyper-parameters in each algorithm are dependent on the data domain and the number of labeled data available.

I tune hyper-parameters in each algorithm using development dataset. The tuned hyper-parameters are summarized in 3.7 with their meaning.

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

First, as I have depicted in section 3.2.2, 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 consistency between the original labels and labels predicted for labeled seeds. I could guess that in a noisy similarity graph, I must ensure the correctness of labeled data (leaders, borrowing a wording in [TC09]) 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. 3.5.2). 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.

3.12 shows the influence of μ_1 in MAD. I firstly set μ_3 as default value and adjusted the value of μ_1. We can see that when the value of μ_1 is small it performs badly. As I have depicted in subsection 3.2.2, the labeled instances are not adjusted to original states in MAD. When the similarity graph includes some noisy edges, as we usually confront, we need to put a high value to μ_1 to keep the consistence between original labels and predicative labels for labeled seeds. We could guess that in a noisy similarity graph, we must ensure the correctness of leaders (labeled data) so that the whole world (classification performance) cannot go out of control (degenerate).

Then I fixed the value of μ_1 to the optimal value and investigated the influence of μ_3, we can see that the performance is more sensitive in the situation of fewer labeled instances from 3.13. One probable explanation is that when the labeled data are not sufficient, the predictive label for vertices in the graph is not reliable. More worse, high-degree vertices (see 3.5) will broadcast mislabeled information to more vertices in the graph. In such kind of circumstance, a higher value should be set to μ_3 so that vertices can become conservative to propagate label to its neighbors.

In many cases in 3.14 and 3.15, I could observe that the performance of SVM and TSVM are sensitive to the change of hyper-parameter C. They perform badly when the value of C is small. The value of hyper-parameter C plays the role of regularization, and underfiting occurs when the value of C is too small while large value increase the risk of overfiting. Therefore, I can find that the optimal values of C is intermediate. Besides we could clearly notice the performance of TSVM is further affected by hyper-parameter p in 3.16. It performs best in the most of situations with p equals to 0.5 due to the fact that the unlabeled data I used during the experiment is nearly balanced in positive and negative sentiment. The optimal value for these hyper-parameters in each domain are summarized in 3.8.

3.6 Summary

To summarize this study

- I evaluate LP on document-level sentiment classification in a resource-scarce language. My method can be applied to any languages in which a small number of labeled reviews are available.

- I run LP with different review representations[1] (content words, phrases, and adjectives), and various similarity measures (dice coefficient, overlap coefficient, Jaccard, cosine similarity) [MS99]. I thereby reveal their impact on the classification performance.

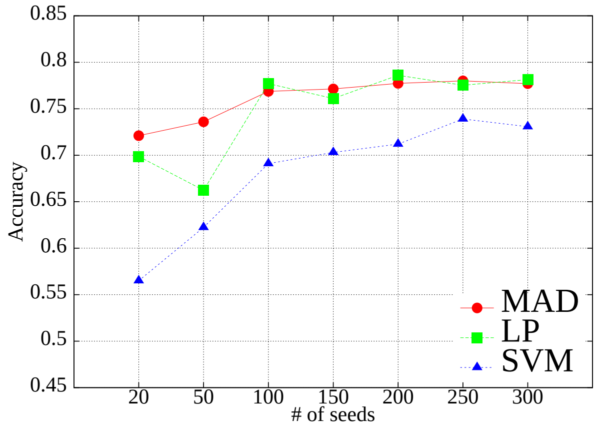
- I compare our method with support vector machines [Vap95] and transductive support vector machines [Joa99], and demonstrate the stability of the classification performance of our method in this task.

- I propose and evaluate the usage of social network analysis metrics including the number of degree, PageRank [PBMW99] value, and the betweenness centrality [Bra01] to locate the initialized seeds so that I could guarantee the good performance.

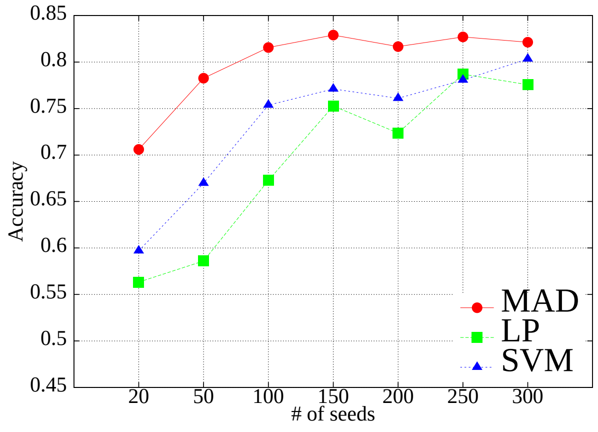
- I also investigate the influence of edge pruning and label propagation control. I conclude that pruning edges does not achieve a similar level of performance like in the choice of seeds, and I can alleviate the side impact caused by the densely connected vertices through explicitly adjust the label propagation.

By now, I adopt special methods and features to resolve the minimally-supervised problem. It is not clear why conventional approaches such as SVM cannot applied in minimally-supervised setting yet. Is there any way to improve them so that they could be applicable when I only have a handful of labeled data? I will provide the answer in my second study.

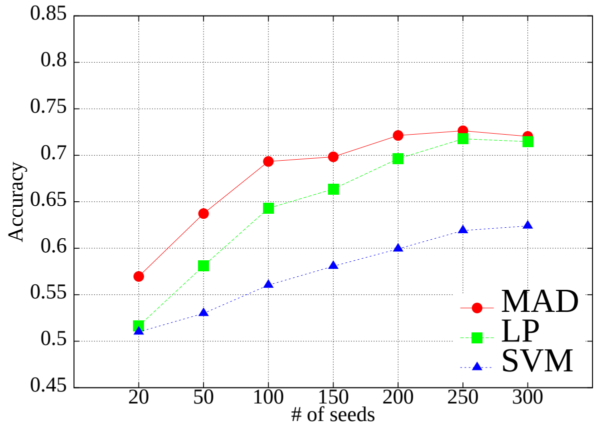
[1]hereafter referred to as sentiment features



(a) Notebook reviews

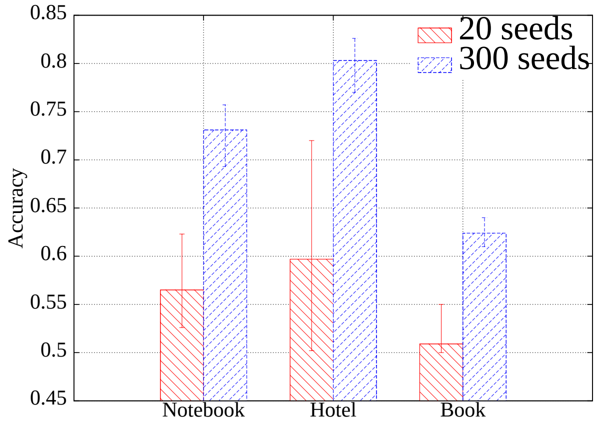


(b) Hotel reviews

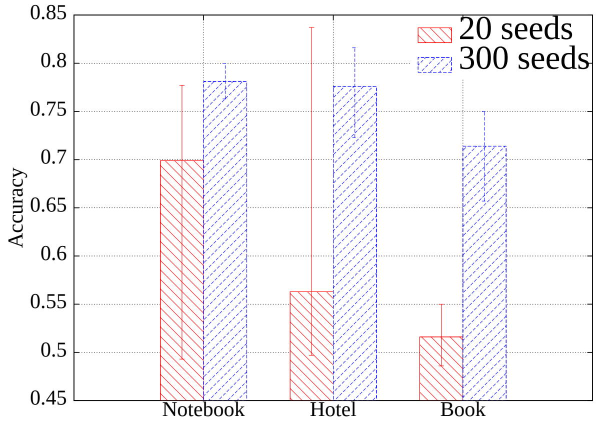


(c) Book reviews

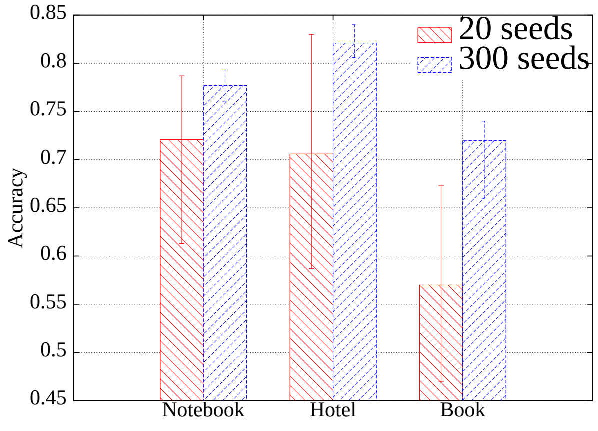
Figure 3.9: Performance comparison



(a) SVM



(b) LP



(c) MAD

Figure 3.10: Performance deviation

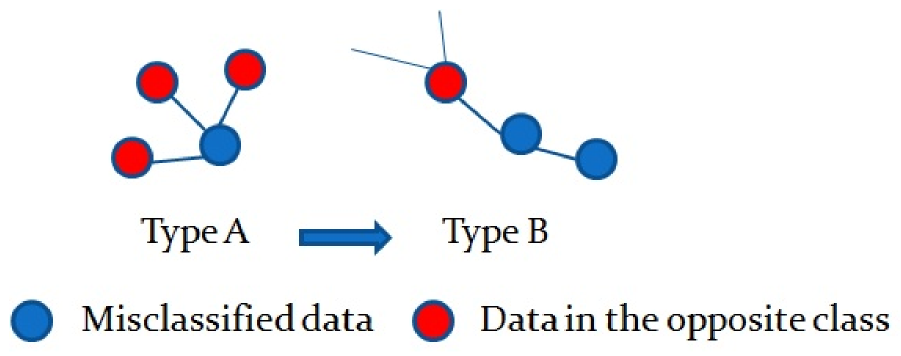
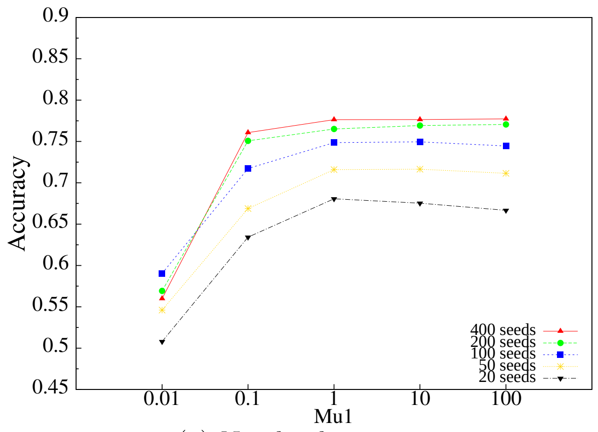
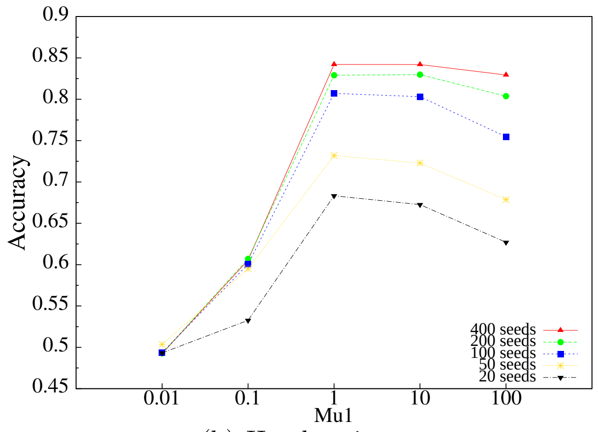


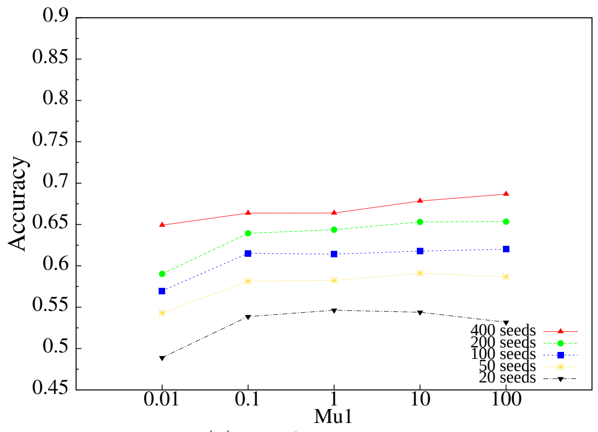
Figure 3.11: Error types in graph-based SSL



(a) Notebook reviews

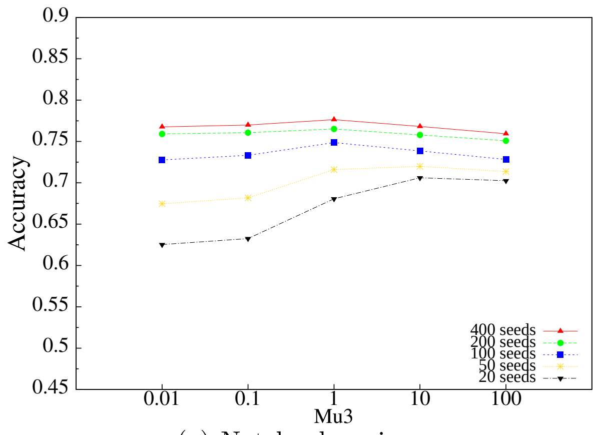


(b) Hotel reviews

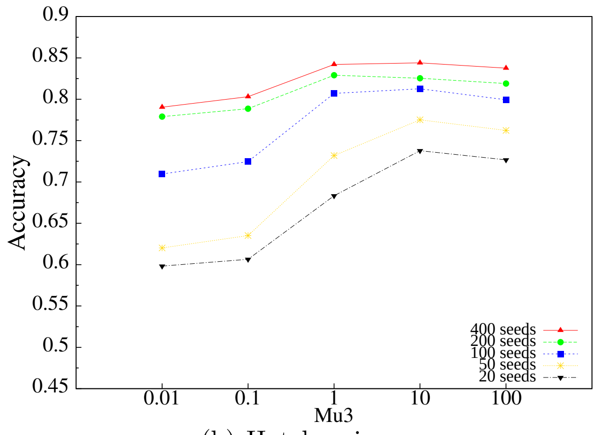


(c) Book reviews

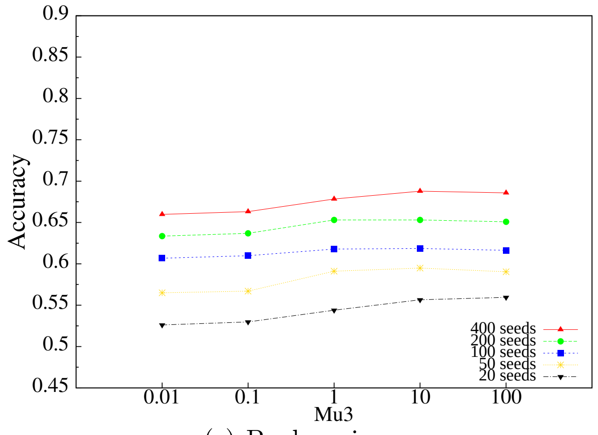
Figure 3.12: Impact of μ_1 in MAD



(a) Notebook reviews

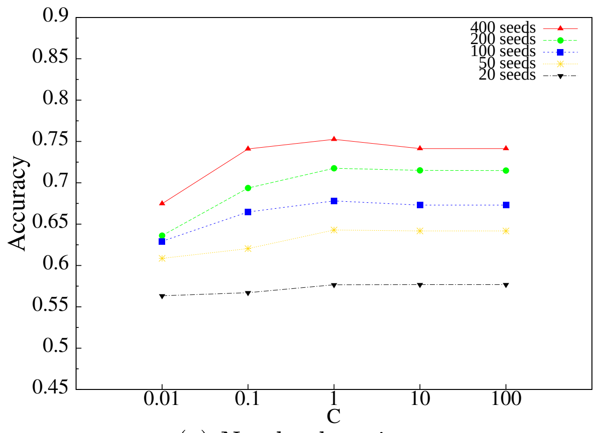


(b) Hotel reviews

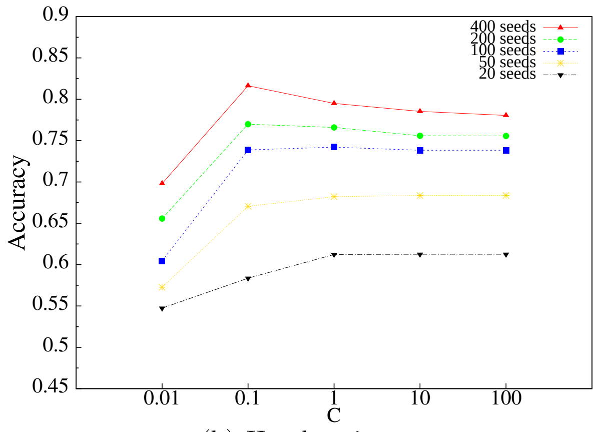


(c) Book reviews

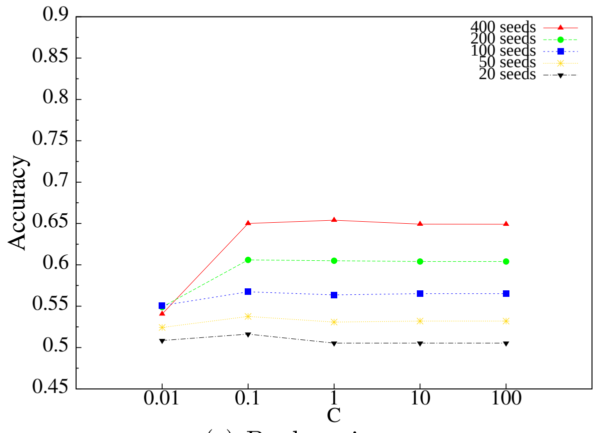
Figure 3.13: Impact of μ_3 in MAD



(a) Notebook reviews

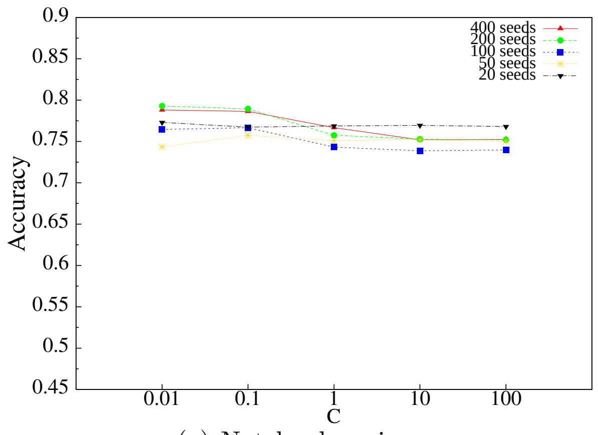


(b) Hotel reviews

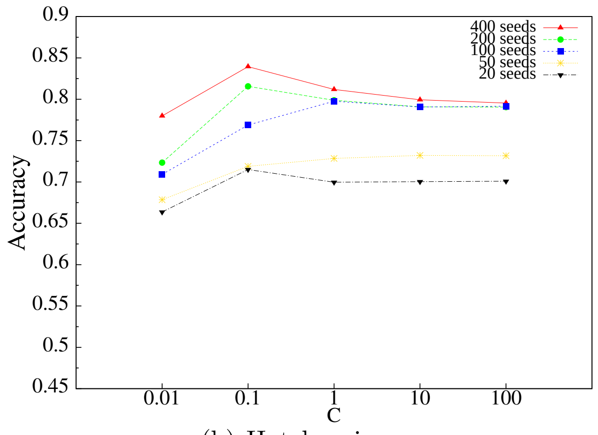


(c) Book reviews

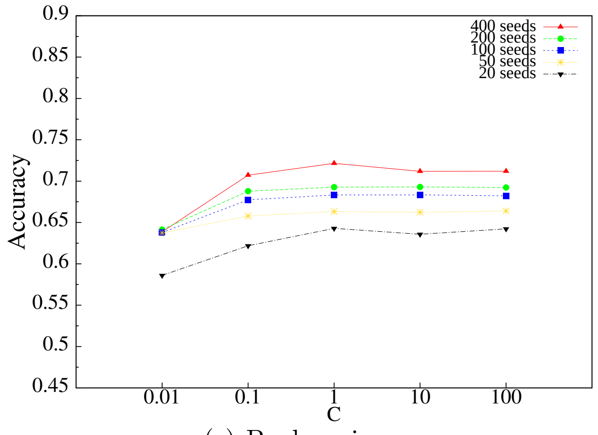
Figure 3.14: Impact of C in SVM



(a) Notebook reviews

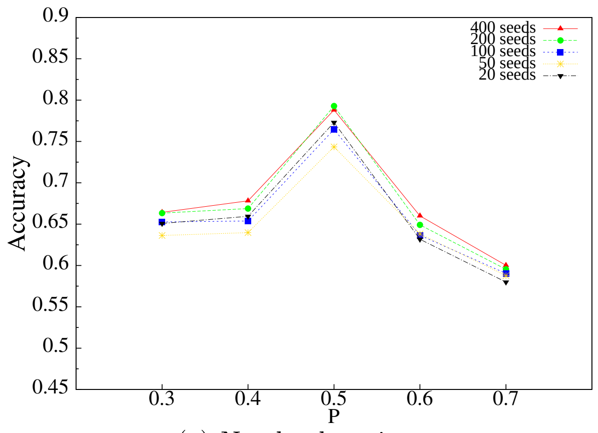


(b) Hotel reviews

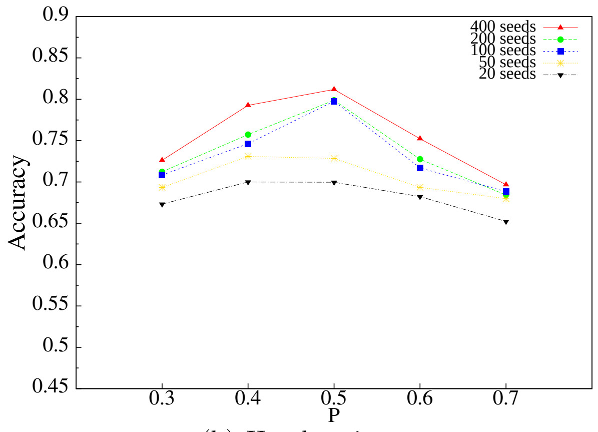


(c) Book reviews

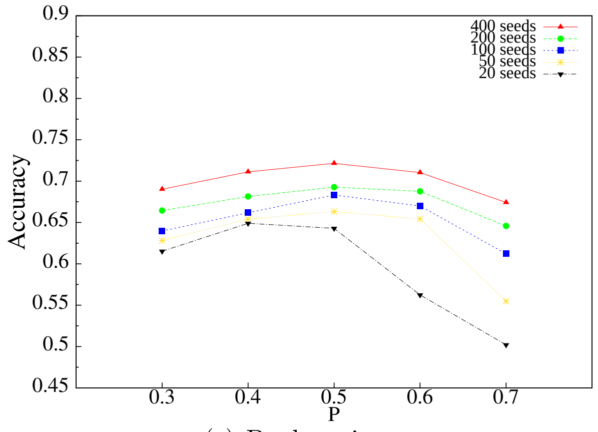
Figure 3.15: Impact of C in TSVM



(a) Notebook reviews



(b) Hotel reviews



(c) Book reviews

Figure 3.16: Impact of p in TSVM

Chapter 4

Task-oriented Word Embedding

4.1 Introduction

Though there are several pioneering studies on predicting readers' feeling after they read the given document[LYC07, TC11], the task of discerning one specific reaction from the others has barely been studied. In this study, I focus on detecting episodes that can trigger horrible feeling from readers which is a emotion classification task on the reader's perspective. I found that such kind of episodes are spreading widely in the consumer generated media (CGM) platforms, and causing bad user experience. For example, the following document (originally in Chinese, but manually translated into English) is a representative horror story in Chinese microblog platform. It was once retweeted more than one thousand times.

> Since people who are living in high-rise apartment usually use elevators, the stairs become the unnoticed place. One night, a girl who lives in the 13th floor wanted to came back home. Unfortunately, the elevator was not in service due to its malfunction. Looking at the long stairs, she was scared. So the girl asked her

> mother come downstairs to pick her up. Her mother came, and the girl went upstairs with her mother. At 12th floor, the girl received the phone call from her mother,'I have just arrived. Where are you?'.

After a brief thinking, readers could realize that the girl in this story was possibly taken away by a ghost other than her mother. I believe that similar stories will arouse bad imagination from many readers, especially, those people who have children. Identifying those episodes in advance can prohibit them touching those improper readers.

In my study, I formulate horror identification as a binary classification problem, and employ widely-adopted supervised learning algorithm, Supported Vector Machine (SVM) [Vap95], to resolve it. Besides conventional bag-of-words (BOW) features (I name them as sparse features in my study), I also investigated the usefulness of features derived from a large scale of unlabeled corpus. Through the empirical evaluation, I confirm that those induced features can improve the performance especially when the number of training data is very small (1% to total target documents). Finally, I propose one novel way to improve the existing word embedding method with task specific supervision.

I conduct experiments on hint fiction, which is a newly emerging literature in microblog platform. Since terrible hint fiction is the typical source that can evoke readers' horrible emotion. I should note that identification of horror episodes is a minimally-supervised classification task, since it is very difficult to accumulate a large amount of horrible stories in practice.

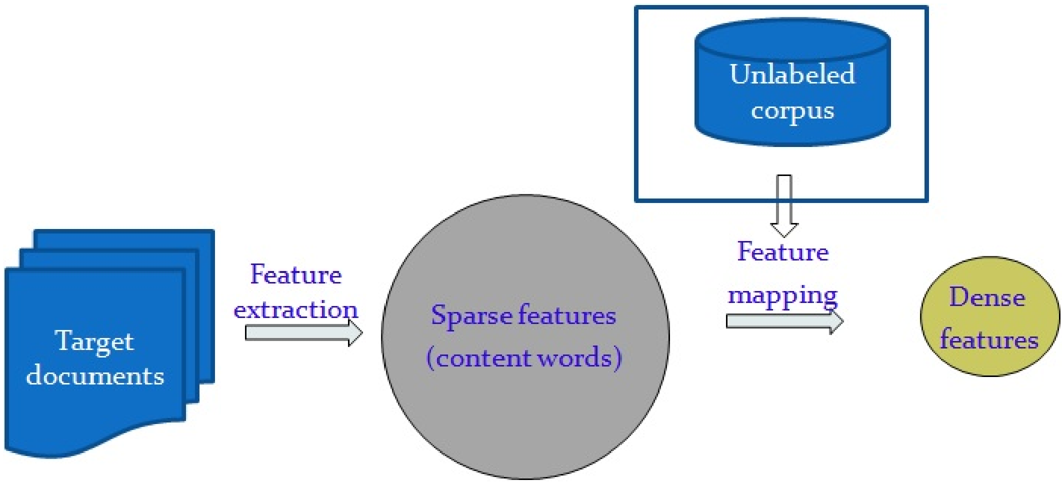


Figure 4.1: Approach overview

4.2 Feature Sparsity Challenge

Usually, supervised classier suppose that there are many common features between training and test data. However, this requirement cannot be met in minimally-supervised setting. Specifically, conventional text classification rely on the content in the documents, for example, unigrams, bigrams and n-grams are commonly adopted features. Due to the power-law distribution of vocabulary (an evidence is represented in section B), it is impossible to include all the content in the training phrase especially when the labeled data is limited. In other words, the classier trained using limited labeled data will encounter many unseen content during test phrase, which cause the classifier cannot make right judgement.

For example, both the following simplified statements

"Her mother killed her father"

and

"The car crashed into the girl"

can arouse terrible feeling among readers. However, classifier trained by simply using one statement cannot recognize the other one as the same category, because there is no common words at all. Mathematically, the inner product between the two corresponding feature vector is zero. Usually, the inner product is the metric to locate the classifier. If we could design an approach to discover the same pattern "Noun hurt Noun" in the examples and encode the pattern into the model training process, we could successfully achieve the classification purpose.

In order to disentangle the commonality among text, I propose the usage of feature mapping derived from unlabelled corpus to densify the sparse feature space. Figure 4.1 depicts the whole procedure of my solution. By leveraging the unsupervised feature learning algorithms, I could gain the feature mapping. Then the BOW features will be transformed into dense features via the resulting feature mapping. Note that we can use the dense features directly to replace ordinal BOW features or combine them with BOW features to form the compound features. I will compare them in the evaluation section 4.5.

In order to gain the feature mapping, I evaluate three unsupervised feature learning methods: Brown-clustering [BDM+92], Latent Dirichlet Allocation [BNJ03] and word embedding [MYZ13] to locate the best approach. In the following section, I will begin with the introduction of them respectively.

4.3 Feature Mapping Approaches

In this section, I first describe the unsupervised learning methods. Then I summarize their characteristics and the demerit. Finally, I give my proposal: task-specific word embedding. The performance comparison will be shown

Figure 4.2: Class-based bigram model in Brown clustering

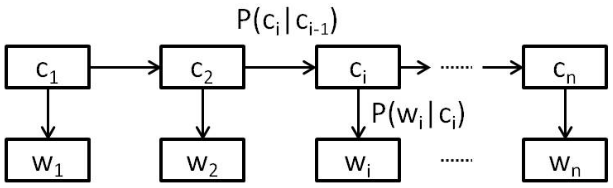
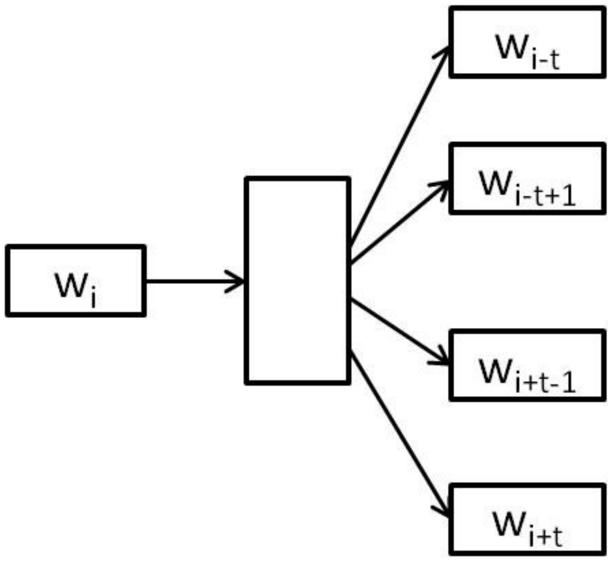


Figure 4.3: Skip-gram model in word2vec



in the experiment section 4.5.

4.3.1 Brown Clustering

Clustering words according to their context (surrounding words distribution) is an ordinary method to obtain the commonality for those words. For example, "father","mother","girl" are clustered into a pronoun group; while "crash" and "kill" are clustered into a transitive verb group[1]. One straight idea is including the cluster IDs of words as auxiliary features, which is also explored in my study. I take measure of the representative hierarchical words

[1]Note that all the groups are specified by ID

clustering algorithm, Brown clustering [BDM+92], in this task. Brown clustering was also exploited in many NLP tasks such as named entity recognition (NER) [MGZ04] and dependency parsing [KCC08].

The Brown-clustering is based on *class-based bigram language model* as 4.2 shows, where w_i is one specific word and c_i specifies the corresponding cluster. We can clearly find that Brown clustering supposes that the cluster of one word is affected by the previous one. More precisely, given one clustering function C ($C(w_i) = c_i$), the quality metric of the function C is defined as follows [Lia05]:

$$\sum_{c,c'} P(c,c') \log \frac{P(c,c')}{P(c)P(c')} + \sum_{w} P(w) \log P(w) \tag{4.1}$$

Formula (4.1) is the actual objective that Brown clustering tries to maximize. Here, c and c' are two consecutive clusters in the text. Interested readers can find the detailed derivation in [Lia05]. Note that the time complexity of Brown clustering is $O(N * C^2)$, where N is the number of words and C is the number of clusters. In other words, it usually take a long time (weeks) to finish the word clustering when we need to deal with a large scale document collection and try to group the words in to fine-grained clusters, for example, thousands of clusters.

4.3.2 Latent Dirichlet Allocation

Latent dirichlet allocation (LDA) [BNJ03] is a well-known unsupervised learning approach, and has various derivatives. It is usually employed to capture the underlining topics in the document. The mechanism is represented in figure 4.4 [2], where θ is the topic distribution for one specific document, z

[2]For simplicity, I omit the introduction of hyper-parameters.

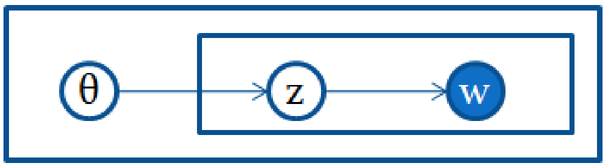


Figure 4.4: LDA mechanism

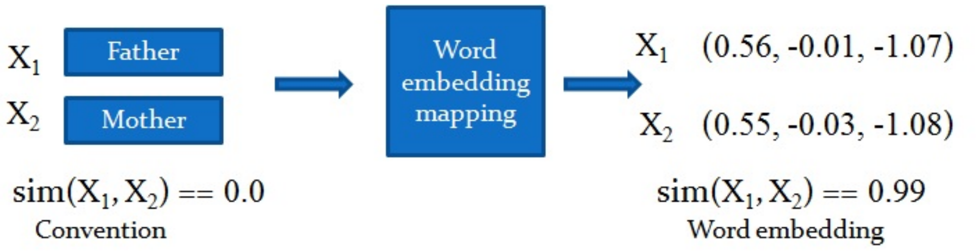


Figure 4.5: Examples using word embedding

is the topic for a word in the document, and w is one specific word. Word w is the only variable we can observe, and the complicated statistical inference is applied to gain the topic distribution. In terms of those two simplified statements given previously in section 4.2, we can get the common topic: “Children” after we apply LDA on them. The derived topic distribution can be taken as features directly.

4.3.3 Word Embedding

Different from the conventional study that assumes that one word correspond to one dimension in the (sparse) feature space, word embedding originated from neural network language model (NNLM) [BSS+06] represents each single word as a dense vector (word feature vector or word vector). Moreover, the representation is induced automatically by an unsupervised method. No feature refinement or devising process exists during the learning process. It is generally believed the word feature vector derived could capture multiple degrees of similarity [MYZ13]. I use word2vec [MCCD13], the state-of-the-art implementation of word embedding, to obtain a vector representation for each word. Two models, continuous bag-of-words model (CBOW) and continuous skip-gram model, are proposed in this framework. Here, I adopt the latter since a better performance has been reported in [MCCD13].

4.3 displays the sketch of continuous skip-gram model. We can clearly observe that the critical point is to predict the words represented using vector form within the specified window around the given word vector w_t. In my evaluation, I set the window size to five ($t = 2$). A commonly adopted soft-max classifier is used during the word vector prediction. One attracting aspect is its capability to conduct the unsupervised learning on the unlabeled corpus with billions of words.

Figure 4.5 shows results after applying word embedding on two real examples ("father" amd "mother"). Note that usually a high dimension word embedding vector is used in reality. We can clearly see the advantage from word embedding: totally different words can has similarity.

Though it is not straightforward compared with the two previous approaches, we can obtain the dense representation with ease: just need to average the word feature vectors for all the content words in the given document. Quite recently, the authors of [LM14] try to derive the vector form for larger unit such as one sentence and even the whole document directly from a large amount of unlabeled corpus. I will consider the application of such kind of methods in my future exploration.

4.3.4 Summary of Previous Approaches

In summary, Figure 4.6 shows the relationship and difference among three feature deriving methods. We can clearly see that they can capture features

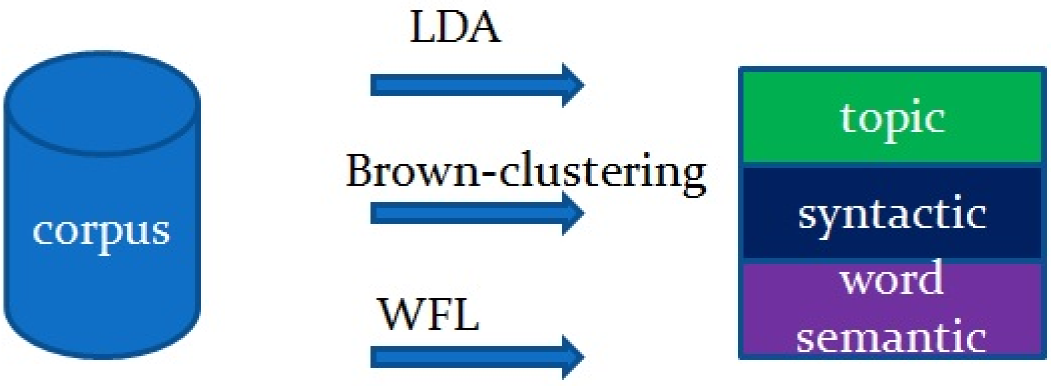


Figure 4.6: Method summary

Table 4.1: Method comparison

Method	Format	Learning context	Complexity
brown clustering	discrete, binary sequence	bi-gram	$O(V * C^2)$
LDA	continuous, probability	document	unknown
word embedding	continuous, real vector	skip-gram	$O(N * D * \log_2(V))$

from different facets, and I will show which one is more helpful for my task in the following section. Moreover, table 4.1 summaries the form of features derived, context for learning features and computational complexity among brown clustering, LDA and word embedding, where N is the size of corpus, D is the dimension of induced vector, V is the vocabulary scale in the corpus, and C is the number of cluster. I should note that the computational complexity are of great importance to utilization of the large scale unlabeled corpus.

We can achieve the purpose to densify the feature space by utilizing those existing approach. However, there is a critical demerit: the lack of task specifics. No matter what kind of tasks we are facing: topic classification, emotion classification even text clustering, the derived features are same; no

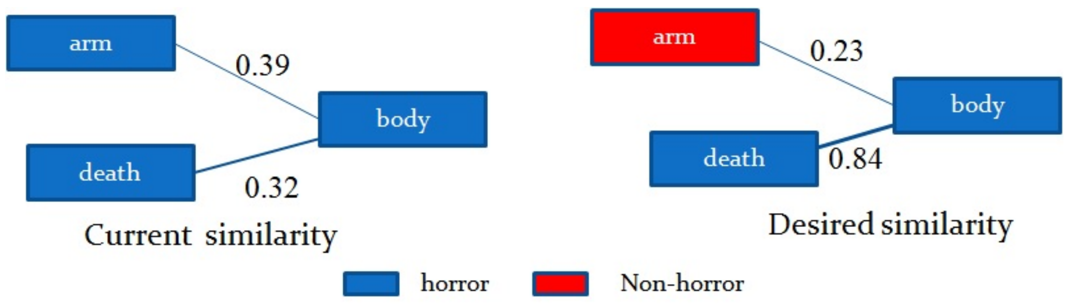


Figure 4.7: Illustration

task characteristics has ever been taken into consideration during the feature learning phrase.

This deficiency could be seen in the illustration in the figure 4.7 (here I only show the results using word embedding, similar phenomenon can also be found in Brown clustering and LDA). Suppose we have three single word documents, and we have labeled the document contains "body" as horrible one. What we really expect is "death" can be classified as horrible document just like the right sub-figure shows, whereas the results we possible get is both documents contain "arm" and "death" will be recognized as horrible documents according to the similarity. In the following section, I will introduce my strategy to integrate task specifics into the existing word embedding vector to achieve the desired results.

4.4 Proposal of Task-oriented Word Embedding

Figure 4.8 depicts my proposal clearly. My proposal is based on existing word embedding, and the reason I choose to extend word embedding is that I have observed better performance (see in section 4.5) using word embedding

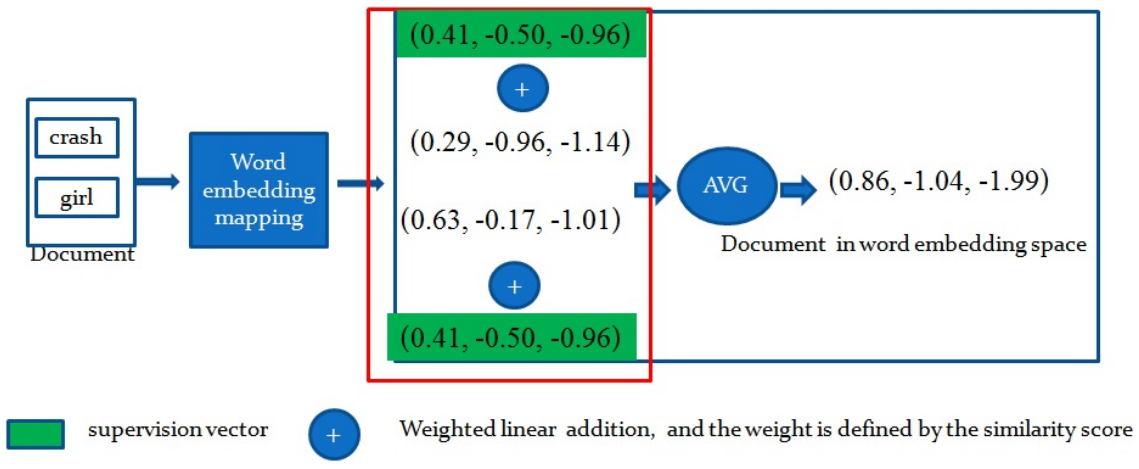


Figure 4.8: Task-oriented word embedding

vectors. I add supervised vectors directly into the existing word embedding through weighted linear addition in word embedding vector space before they are averaged to form the final vector for the document, and the weight is calculated by using the similarly score between target word embedding vector and supervised vector specified, and I should note that the weight could be negative. We can consider the supervised vectors as anchors for specific task, and then adjust the other vectors according to the similarity. Note that the supervised vectors are important words for the task. Next, I will introduce two ways to gain the supervised vectors.

The first way to gain the supervised method is heuristic. I manually specify three words “dead body”, “sudden” and “found” as supervision. The rational is that they are horror related words so they should be helpful for horror identification. I should mention that this manual participation is optional. I should also note that all these important words will be transformed into word embedding vectors, and there could be also other methods to gain the supervised vectors.

The second method is automatic. There are three steps:

- Firstly, locate the distinguishing words in horrible documents and un-horrible documents respectively, and rank them according the frequency. Only top-10 words appeared in horrible documents are considered as important words and keep top-50 words in un-horrible as filtering reference.

- Measure the similarity score between the candidates and filtering reference, and set the filtering threshold as 0.1. I record the number below the threshold for each candidate and take this number as selecting criterion.

- Locate the top-5S candidates whose criterion is below than 10 as final important words.

There are two things that worth explaining especially:

Firstly, the reason that I do not use conventional feature selection methods as such Chi-square or the co-efficient value from SVM is that when we are in the minimal-supervised setting, the resulting critical features will also become unreliable due to limited training data.

Secondly, the necessity of the second and third step is that eventually I will feed the corresponding word embedding vector into the classifier. As we have seen in section 4.3.3, even two totally different words could have high similarity. The purpose of the second and third steps is to avoid bring noise for the classifier.

Table 4.2: Statistics on dataset

Class	# of documents	# of content words
Horror	1,008	13,260
Humor	1,114	14,876
Moved	1,018	12,674

4.5 Evaluation and Discussion

4.5.1 Setting

In this study, the target domain is hint fiction, which is a newly emerging form of literature in microblog platform. I collected hint fictions from the website[3] which lists various genres of hint fictions. I only kept three types: horror, humor, and moved, since they are closely related to reader emotion. Table 4.2 report the statistics on the dataset. We can find that the average number of words in each text is around 130, which conforms to length limit in microblog platform. I treat humor and moved types as non-horror in my study.

I randomly separated each type of those text into five equally-sized parts, and then combine them accordingly. Eventually, there are five subsets and each of them are nearly balanced in the classes. In this evaluation, I take those humor and moved text as non-horror instances. I employ five-round evaluation and report average precision, recall and F1 with the optimal hyper-parameter setting. Note that in each round I only use one subset as training data (20%), and treat the other four subsets (80%) as test data. The motivation of this strategy locates in the fact that labeling data is time consuming and labor intensive, and we cannot guarantee the number of training data

[3]http://v.gxdxw.cn/

Table 4.3: Performance using textual features

Feature	Precision	Recall	F1
unigrams	0.760	0.442	0.56
unigrams and bigrams	0.892	0.250	0.388

available is surely more than the test counterpart in practice. Moreover, I control the ratio of labeled data from 1% to 20% so that I could find the performance changes with the number of supervision.

The unlabeled corpus is provided by datatanng[4]. It consists of more than 10.5 million Chinese web text, 7.36 million content words after pre-processing. The time spans in time from 1992 to 2011.

In this study, I use the optimized version[5] as the implementation of brown clustering. Hoffman's online LDA [HBB10] is the used to conduct the LDA topic model training. In terms of word embedding, I make use of the implementation provided by Google[6]. I adopt the Gensim version used[7] when I carried out the reference, since it is convenient to customize feature concatenation which will be introduced in the section 4.5.3.

4.5.2 Textual Feature Exploration

Firstly, I make use of unigrams and unigram&bigrams to detect horrible stories. Table 4.3 shows the results when training data is 20%. I take this as the optimal performance of conventional text based classification in minimally supervised setting. We can clearly find that although I can get good performance in terms of precision, the recall is very bad. In other words, a

[4]www.datatang.com

[5]https://github.com/percyliang/brown-cluster

[6]https://code.google.com/p/word2vec/

[7]http://radimrehurek.com/gensim/models/word2vec.html

Table 4.4: Top ten significant features

Rank	Words	Relation to horror
1	elevator	place
2	dead body	object
3	found	action
4	eyes	object
5	voice	object
6	mirror	object
7	midnight	time
8	wife	subject
9	boy	subject
10	suddenly	time

lot of terrible stories are missed due to the limited training data and feature sparsity. Moreover, combining bigrams make recall become worse, and the explanation is that the feature space become more sparse. In the following evaluation, I only keep unigrams as conventional classifying features.

In order to get the better understanding on those derived features, I conduct feature analysis. It is known that the features are not equal in distinguishing instances. Here, I use the conventional Chi-square feature selection [SMG10] to examine significance of textual features.

Table 4.4 lists the top-10 significant features and their relation to the horrible storylines. Even the feature selection I use is conventional and simple, I can see the amazing association between the significant features and the terrible scenes. For example the authors of horrible hint fiction tend to establish the happenings to stairwell or night. And “found” play critical role in discriminating horrible stories from the others. It is used to describe things

Table 4.5: Top ten secondary features

Rank	Words	Meaning
1	DNA	proper noun
2	together	adverb
3	for a while	adverb
4	angrily	adverb
5	thirties	special times
6	god	proper noun
7	make the decision	verb
8	start	verb
9	droop	verb
10	context	noun

occur unexpectedly. Moreover, nouns are helpful to judge whether the text is terrifying or not. One possible explanation is that the thematic information in terrible stories are different from the other genres.

Comparatively, table 4.5 lists the bottom features (here, I named them as to "secondary features") judged by using Chi-square score, and there is no clear clues between those bottom features and the terrible episodes.

By now, we have realized the challenge caused by feature sparsity: a lot of horrible documents will be ignored since many their features are never found in training phrase.

4.5.3 Usefulness of Induced Features

In this section, I will show the benefit we can get from those induced features. Firstly, I will compare the two ways to leverage induced features. Then, I will introduce a series of evaluation including the impact of the size of labeled

Table 4.6: Usage comparison(labeled data: 1%)

Method	Precision	Recall	F1
Replacement	0.512	0.568	0.534
Compound	0.656	0.306	0.398

Table 4.7: Usage comparison(labeled data: 20%)

Method	Precision	Recall	F1
Replacement	0.656	0.602	0.626
Compound	0.716	0.632	0.670

data, the size of unlabeled corpus for feature learning, the dimension of word embedding vector. Next I will present the effectiveness of task-oriented word embedding. Finally, I explore the semantics captured by word embedding.

I carried out the evaluation in the framework of binary classification, and used the widely-adopted supervised learning algorithm SVM [Vap95] in this task. I specify the number of clusters for Brown-clustering to 100, and set the dimension of LDA topic vector and word embedding to 100.

Comparison on Utilization of Induced Features

As I mentioned previously in section 4.2, there are two ways to use the dense features. I make comparison on them with 1% and 20% labeled data respectively. Table 4.6 and table 4.7 show the results on using word embedding feature vectors. We can conclude that when the number of labeled data is very small (here 1%) we should adopt the replacement way, while combination of BOW features and word embedding features is a better way when a certain number of labeled data (here 20%) is available. Without special description, the following evaluations are conducted using only 1% labeled

Table 4.8: Impact of dimension with %1 training

Dim of induced vector	Precision	Recall	F1
10	0.490	0.264	0.316
100	0.512	0.568	0.534
1000	0.598	0.528	0.552

Table 4.9: Impact of dimension with %20 training

Dim of induced vector	Precision	Recall	F1
10	0.792	0.538	0.640
100	0.764	0.614	0.680
1000	0.736	0.642	0.684

data since my focus is minimally-supervised setting.

Usefulness of Induced Feature

Here, I conduct the experiment to investigate the usefulness of those induced features derived from the unlabeled corpus. Figure 4.9 presents the performance change with the number of labeled data. We can clearly see that the performances of all the methods except LDA become better when more labeled data is added. More important, word embedding feature vectors are very helpful to improve the recall and F1 when the number of labeled data is very small, I owe this to the . At last, word embedding outperform brown clustering and LDA which shows that word semantics are more helpful than syntax and topics in detecting horrible stories. The explanation is that horror identification needs to understand semantics of the stories.

Figure 4.10 shows the performance change with the number of unlabeled corpus for feature learning. We can conclude that conclude that we can ben-

Table 4.10: Performance comparison

Method	Precision	Recall	F1
BOW	0.810	0.014	0.026
Word embedding	0.598	0.528	0.552
Proposal with automatic supervision	0.622	0.570	0.588
Proposal with customized supervision	0.616	0.596	0.596

efit from more unlabeled corpus since we can gain more axillary information.

In terms of word embedding, I also investigate the impact of dimension of feature vector. Table 4.8 and table 4.9 list the results, where we could see that high dimension vector is favored in applying word embedding.

4.5.4 Effectiveness of Task-oriented Word Embedding

Table 4.10 lists the performance comparison. Here, I keep the dimension of induced to 1000. We can conclude that my proposal is effective to improve the performance owing to the supervised information added.

Instance Analysis

I conduct the instance analysis so that we could have a better understanding on the challenges caused by feature sparsity. Figure 4.11 shows an illustrative example, the left one is expected classier training from sufficient labeled data while the right one is the bad suited classifier due to limited labeled data. We can intuitively found that there is obvious difference between the desired classifier and bad classifier, and the bad suited classifier cannot perform well due to the limited labeled samples provided in the training phrase. We cannot expect that only a few training data can reflect the whole view of the data distribution.

Next, I will present the analysis based on real data. The following paragraphs exhibit three representative training samples from one real group of labeled horrible documents used in my study. For better understanding, I manually translated the text in English.

- 昨夜$_1$ 我$_2$ 开$_3$ 出租车$_4$ ，拉着$_5$ 五个$_6$ 人$_7$ 去某村$_8$，两个$_9$ 女的$_{10}$ 穿$_{11}$ 白衣服$_{12}$ 三个$_{13}$ 男士$_{14}$ 穿$_{15}$ 黑衣服$_{16}$。不久$_{17}$ 某村$_{18}$ 就到了$_{19}$。一个$_{20}$ 白衣$_{21}$ 女士$_{22}$ 付完$_{23}$ 车费$_{24}$，我$_{25}$ 就回家$_{26}$了 。早晨$_{27}$ 想吃$_{28}$ 早点$_{29}$ 拿出$_{30}$ 钱$_{31}$ 一看$_{32}$ 是冥币$_{33}$。气得$_{34}$ 我$_{35}$ 开车直奔$_{36}$ 某村$_{37}$。来到$_{38}$ 那家门口$_{39}$ 压住$_{40}$ 怒气$_{41}$ 敲门$_{42}$。一$_{43}$ 农妇$_{44}$ 开门$_{45}$，我$_{46}$ 便说$_{47}$ 大姐$_{48}$，你家$_{49}$ 昨夜$_{50}$ 来的$_{51}$客人$_{52}$呢？农妇$_{53}$ 说$_{54}$ 没来$_{55}$ 客人$_{56}$啊，俺家$_{57}$ 老母猪$_{58}$ 昨夜$_{59}$ 生$_{60}$ 小猪崽$_{61}$了，来到$_{62}$ 猪舍$_{63}$ 一看$_{64}$，三黑$_{65}$ 两白$_{66}$ 一点不假$_{67}$ 。

  (Yesterday night$_1$ I$_2$ drove$_3$ a taxi$_4$ to take$_5$ five$_6$ passengers$_7$ to a village$_8$, two$_9$ female passengers$_{10}$ wore$_{11}$ in while$_{12}$ and three$_{13}$ male passengers$_{14}$ wore$_{15}$ in black$_{16}$. Soon$_{17}$ we arrived$_{19}$ the village$_{18}$. One$_{20}$ woman$_{22}$ who wore in white$_{21}$ paid$_{23}$ the fare$_{24}$,then I$_{25}$ went home$_{26}$. Next morning$_{27}$, I took$_{30}$ the money$_{31}$ to buy$_{28}$ the breakfast$_{29}$ and realized$_{32}$ that the money is false paper money burned as an offering to the dead$_{33}$. I$_{35}$ got angry $_{34}$and drive directly$_{36}$ to that village$_{37}$. When I came$_{38}$ to that house$_{39}$, I suppressed$_{40}$ my anger$_{41}$ to knock out the door$_{42}$. A$_{43}$ countrywoman$_{44}$ opened the door$_{45}$. I$_{46}$ said$_{47}$ to her "Elder sister$_{48}$, where are the guests$_{52}$ came$_{51}$ to your house$_{49}$ yesterday night$_{50}$?" She$_{53}$ answered$_{54}$ that there was no$_{55}$ guests$_{56}$, but the sow$_{58}$ of her family$_{57}$ gave birth to$_{60}$ little pigs$_{61}$ yesterday night$_{59}$. When I came$_{62}$ to the pig house$_{63}$, I found$_{64}$ that there are really$_{67}$ three of them are black$_{65}$ and the other two are white$_{66}$.)

- 王$_1$ 发现$_2$ 一个$_3$ 诡异$_4$得令人$_5$ 毛骨悚然$_6$ 的事$_7$，每次$_8$ 他$_9$ 回家$_{10}$走到$_{11}$ 那个路口$_{12}$ 都是$_{13}$ 红灯$_{14}$，无论$_{15}$是白天黑夜$_{16}$，或早或晚$_{17}$，只要$_{18}$ 是他$_{19}$ 按正常的$_{20}$ 速度$_{21}$ 走到$_{22}$那个路口$_{23}$，没有迟疑$_{24}$，没有犹豫$_{25}$，一定$_{26}$ 是红灯$_{27}$。一次$_{28}$，他$_{29}$ 下定决心$_{30}$ 一定要$_{31}$ 看到$_{32}$ 绿灯$_{33}$。于是$_{34}$ 他$_{35}$ 走到$_{36}$ 离$_{37}$ 那个路口$_{38}$ 还有$_{39}$ 一段距离$_{40}$ 的时候$_{41}$ 停了下来$_{42}$，等了$_{43}$ 几十秒钟$_{44}$，然后$_{45}$ 慢慢$_{46}$ 往$_{47}$ 路口$_{48}$ 走去$_{49}$。果然$_{50}$ 是绿灯$_{51}$。他$_{52}$ 嘴角$_{53}$ 微扬$_{54}$，然后$_{55}$ 大步走过$_{56}$ 路口$_{57}$。突然$_{58}$，他的$_{59}$ 侧面$_{60}$ 响起$_{61}$ 一阵刺耳$_{62}$的刹车声$_{63}$，声音$_{64}$ 越来越近$_{65}$，然后$_{66}$ 地上$_{67}$ 一片血红$_{68}$。

  (Mr Wang$_1$ found$_2$ a$_3$ strange$_4$ and and make him feel$_5$ creepy$_6$ thing$_7$, every time$_8$ when he$_9$ went home$_{10}$ to pass$_{11}$ the intersection$_{12}$, no matter$_{15}$ is daytime or night$_{16}$, whenever$_{17}$, as long as$_{18}$ he$_{19}$ arrived$_{22}$ at the intersection$_{23}$ in usual$_{20}$ speed$_{21}$ without hesitate$_{24}$ and without shilly-shally$_{25}$, the traffic light was always$_{13}$ red$_{14}$. One time$_{28}$, he$_{29}$ determined$_{30,31}$ to watch$_{32}$ the green light$_{33}$. So$_{34}$ he$_{35}$ stopped$_{42}$ when$_{41}$ there was some distance$_{40}$ to the intersection$_{38}$, and waited$_{43}$ dozens of seconds$_{44}$, then$_{45}$ he walked$_{49}$ slowly$_{46}$ underlineto$_{47}$ the intersection$_{38}$. It was really$_{50}$ green light$_{51}$, and he was happily$_{53,54}$ to pass$_{56}$ the intersection$_{57}$. Suddenly$_{58}$, a ear-piercing$_{62}$ break sound$_{63}$ occurred$_{61}$ at his$_{59}$ side$_{60}$, and the sound$_{64}$ became nearer and near$_{65}$, then$_{66}$ there was blood-red$_{68}$ on the ground$_{67}$.)

- 杜$_1$ 和$_2$ 我$_3$ 一起$_4$ 长大$_5$ ，杜$_6$ 长的帅$_7$ ，有$_8$ 很多$_9$ 女友$_{10}$ ，都$_{11}$ 很漂亮$_{12}$。他$_{13}$ 和$_{14}$ 我$_{15}$ 读医$_{16}$，解剖课$_{17}$ 让$_{18}$ 我$_{19}$ 吐了$_{20}$ 多回$_{21}$，他$_{22}$ 是$_{23}$ 高才生$_{24}$，有$_{25}$ 自己的$_{26}$ 工作室$_{27}$。快考试了$_{28}$，我$_{29}$ 找他$_{30}$ 借$_{31}$ 笔记$_{32}$，他$_{33}$ 不在$_{34}$，我$_{35}$ 去$_{36}$ 工作室$_{37}$ 找他$_{38}$，门$_{39}$ 没锁$_{40}$，我$_{41}$ 好奇地$_{42}$

探头去看$_{43}$，巨大的$_{44}$ 药水缸$_{45}$ 内泡着$_{46}$ 的多具$_{47}$ 女$_{48}$ 尸体$_{49}$，那么$_{50}$ 美丽$_{51}$ 的容颜$_{52}$，啊！他$_{53}$ 睁大$_{54}$ 眼睛$_{55}$ 出现$_{56}$ 我$_{57}$ 眼前$_{58}$，你$_{59}$ 知道$_{60}$的太多了$_{61}$⋯

(Mr Du$_{1}$ grew up$_{5}$ together with$_{2,4}$ me$_{3}$, he$_{6}$ was handsome$_{7}$ and had$_{8}$ many$_{9}$ girlfriends$_{10}$. Both of us$_{13,14,15}$ majored in medical science$_{16}$. The dissect course$_{17}$ make$_{18}$ me$_{19}$ spite$_{20}$ many times$_{21}$. He$_{22}$ was$_{23}$ a top student$_{24}$ and owned$_{25}$ his own$_{26}$ studio$_{27}$. When the examination came$_{28}$, I$_{29}$ looked for him$_{30}$ to borrow$_{31}$ the notebook$_{32}$, but he$_{33}$ was absent$_{34}$. I$_{35}$ went$_{36}$ his studio$_{37}$ to find him$_{38}$ and found that the door$_{39}$ was unlocked$_{40}$. I$_{41}$ was curious$_{42}$ to go into the studio$_{43}$. There was a huge$_{44}$ jar full of medical liquid$_{45}$ and there were$_{46}$ many$_{47}$ female$_{48}$ dead body$_{49}$, so$_{50}$ beautiful$_{51}$ faces$_{52}$! He$_{53}$ appeared$_{56}$ in front of me$_{57,58}$ open-eyed$_{54,55}$, you$_{59}$ know$_{60}$ too much$_{61}$...)

In a summary, these statements are about ghost (sample 1), accident (sample 2) and killing (sample 3) scenes which can trigger terrible feeling for readers.

One representative horrible documents predicted by the classifier trained using those labeled horrible documents is listed as follows. Though it is similar to the training sample 1 in terms of the ghost scene, the reason why it can be judges as a horrible story is that there are several common words such as “dead body” and “found” . I have found that only the documents that share the common words with labeled documents could be judged as possibly horrible documents. In minimally-supervised setting, totally, only 6 such kind of horrible stories can be located. Put it simply, the behavior of the classifier trained from limited labeled data will degrade as a dictionary look-up.

- 他$_{1}$ 睡了$_{2}$ 许久$_{3}$，醒来$_{4}$ 发现$_{5}$ 自己$_{6}$ 在$_{7}$ 一个$_{8}$ 陌生的$_{9}$ 房间$_{10}$。房间$_{11}$的窗户$_{12}$ 很大$_{13}$，外面$_{14}$ 是一片光$_{15}$。渐渐$_{16}$ 他$_{17}$ 发觉$_{18}$ 房里$_{19}$ 有$_{20}$股难闻的$_{21}$ 味道$_{22}$，是腐朽的$_{23}$ 气味$_{24}$。他$_{25}$ 来到$_{26}$ 窗边$_{27}$，奋力$_{28}$ 推开$_{29}$ 窗户$_{30}$ 呼吸着$_{31}$，他$_{32}$ 没有$_{33}$ 注意$_{34}$ 自己的$_{35}$ 肌肤$_{36}$ 在空气下$_{37}$ 已慢慢$_{38}$ 被灼烧$_{39}$。没有痛$_{40}$，没有鲜血$_{41}$，直到$_{42}$ 他$_{43}$ 露出$_{44}$ 森森白骨$_{45}$。故宫博物院$_{46}$，一具$_{47}$ 最新$_{48}$ 发现的$_{49}$ 保存$_{50}$ 完好的$_{51}$ 古尸$_{52}$，一夜之间$_{53}$ 腐烂$_{54}$。存放$_{55}$ 尸体$_{56}$的玻璃馆$_{57}$上裂了$_{58}$ 一道口子$_{59}$。

  (He$_{1}$ has slept$_{2}$ for a long time$_{3}$. When he wake up$_{4}$, he found$_{5}$ that himself$_{6}$ was$_{7}$ in a$_{8}$ strange$_{9}$ room$_{10}$. The window$_{12}$ of the room$_{11}$ is very big$_{13}$, and the outsides$_{14}$ is full of light$_{15}$. Gradually$_{16}$, he$_{17}$ realized$_{18}$ that there was$_{20}$ unpleasant$_{21}$ smell$_{22}$ in the room$_{19}$, and it was rotten$_{23}$ smell$_{24}$. He$_{25}$ went$_{26}$ to the window$_{27}$ and struggled$_{28}$ to open it$_{29,30}$ and breathed$_{31}$. He$_{32}$ did not$_{33}$ notice$_{34}$ that his own$_{35}$ skin$_{36}$ was burn$_{39}$ slowly$_{38}$ in the air$_{37}$. Without pain nor blood$_{40,41}$, his bone appears$_{43,44,45}$. In the National Palace Museum$_{46}$, a$_{47}$ newly$_{48}$ found$_{49}$ mummy$_{52}$ with perfect$_{51}$ preservation$_{50}$ became rotten$_{54}$ overnight$_{53}$. There was a hole$_{58,59}$ in the glass coffin$_{57}$ that used to store$_{55}$ the dead body$_{56}$.)

The following horrible instances are extracted from the statements correctly judged (total number: 336) by adopting word embedding vectors. It is very representative since is no common words at all between them and the training samples. Even the topic is also ghost related, it cannot be recognized by supervise model training on bag-of-words (BOW) features.

- 一位$_{1}$ 跑夜班的$_{2}$ 出租车$_{3}$ 司机$_{4}$ 夜里$_{5}$ 生意$_{6}$ 非常好$_{7}$，有一天$_{8}$ 他妻子$_{9}$ 为他$_{10}$ 求了$_{11}$ 一个$_{12}$ 平安符$_{13}$，之后$_{14}$ 晚上$_{15}$再也$_{16}$拉不到$_{17}$客$_{18}$了。

(The business$_{6}$ of one$_{1}$ night$_{2}$ taxi$_{3}$ driver$_{4}$ was very good$_{7}$. One day$_{8}$, his wife$_{9}$ gave him$_{10,11}$ a$_{12}$ talisman$_{13}$, after that$_{14}$ there was no clients$_{17,18}$ anymore$_{16}$ in the evening$_{15}$.)

The last instance are the horrible stories judged (total number: 373) by using my proposal. There is no common words between it and the training samples either, and it cannot be located by using existing word embedding vectors.

- 厕所$_{1}$ 上到一半$_{2}$，脑袋$_{3}$ 上感到$_{4}$ 一阵冰凉$_{5}$，伸手$_{6}$ 一摸$_{7}$，貌似$_{8}$ 是血$_{9}$。随即$_{10}$ 抬头$_{11}$ 一望$_{12}$，一个$_{13}$ 披头散发$_{14}$ 苍白$_{15}$ 脸孔$_{16}$，搭配$_{17}$ 红色$_{18}$ 液体$_{19}$的脑袋$_{20}$ 倒挂在$_{21}$ 头上方$_{22}$，见他$_{23}$ 望向自己$_{24}$，咧嘴一笑$_{25}$，状似狰狞$_{26}$。

  (When the man in the toilet$_{1,2}$, he had a cold feeling$_{4,5}$ in his head$_{3}$. He held out his hand$_{6}$ to touch it$_{7}$, and it looked like$_{8}$ blood$_{9}$. Immediately$_{10}$ he raised his head$_{11}$ to look at it$_{12}$ and found a$_{13}$ pale$_{15}$ face$_{16}$ with dishevelled hair$_{14}$ accompanying$_{17}$ a head$_{20}$ with red$_{18}$ fluid$_{19}$ hang upside down$_{21}$ on the top$_{22}$. When the head know the man$_{23}$ is looking at her$_{24}$, she grinned$_{25}$ and became mean and ferocious$_{26}$.)

Inspection on Word Embedding

Though there is study [MYZ13] claimed that word embedding can capture word semantics, it is not clear what kind of exact semantics can be captured for Chinese language. Here, I try to investigate the semantics captured using word embedding. Firstly, I cluster the words in the word embedding space according to the similarity score. Figure 4.12 represents parts of the clustering results, and we can clearly find the word embedding can capture context semantic well.

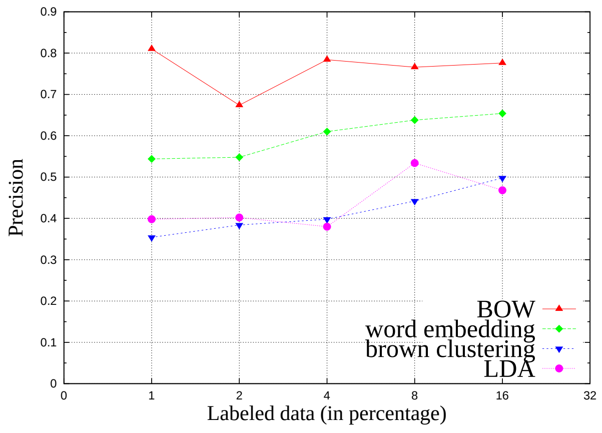
Next, I take advantage of a Chinese synonym dictionary [8] to analyze the word embedding semantics. Figure 4.13 show the results, and we can observe that low dimension word embedding feature can capture synonymy information very well.

4.6 Summary

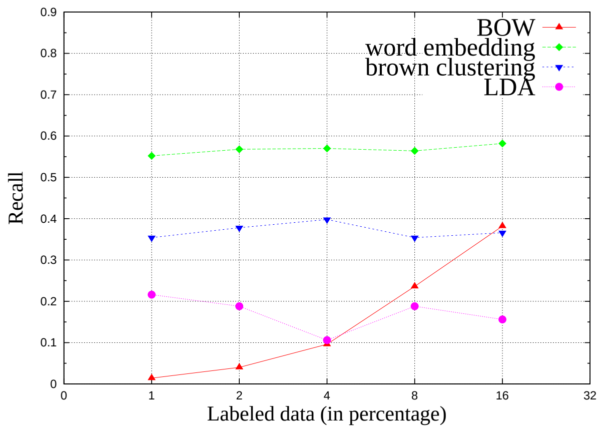
- In order to address the feature sparsity challenge, we exploit the utilization of induced vector from a large scale unlabeled corpus. Three kinds of approaches including Brown-clustering [BDM+92], Latent Dirichlet Allocation [BNJ03] and word embedding [MYZ13] are evaluated and compared in the task of horrible stories identification. We found vectors derived using word embedding can outperform the ones from the other two methods, but the distinguishing role of conventional textual features is irreplaceable.

- I propose one way to integrate decerning capability of textual features into the induced feature vector space. Through the comprehensive empirical study, we demonstrate that our approach is promising to augment task specificity which is ignored by neural network langue model training.

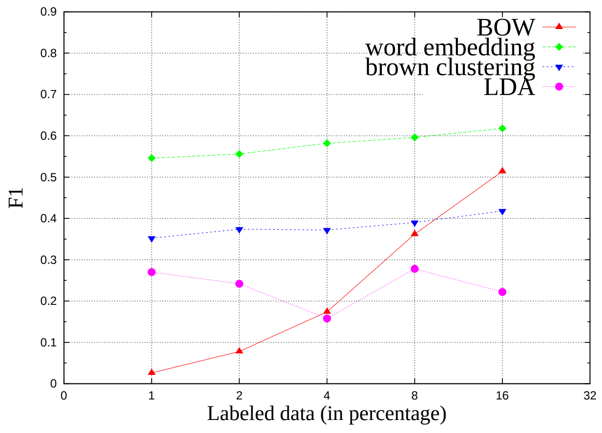
[8]http://www.ltp-cloud.com/download/



(a) Precision

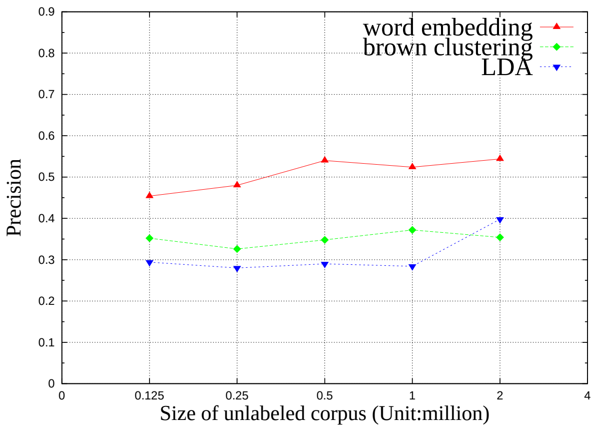


(b) Recall

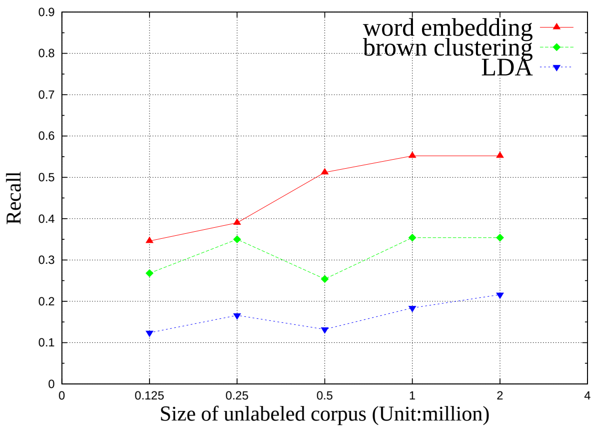


(c) F1

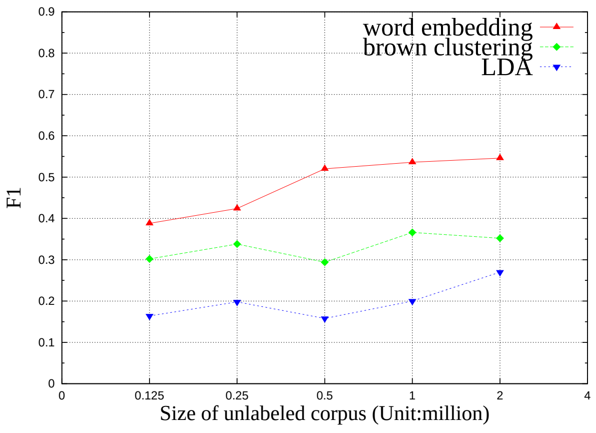
Figure 4.9: Impact of labeled data



(a) Precision



(b) Recall



(c) F1

Figure 4.10: Impact of unlabeled corpus

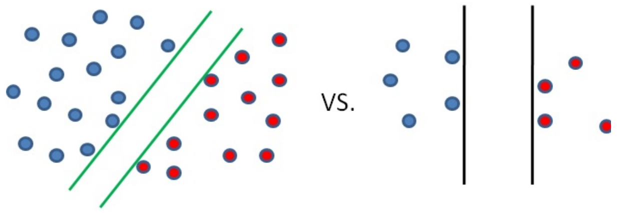


Figure 4.11: Bad suited classifier due to limited training data

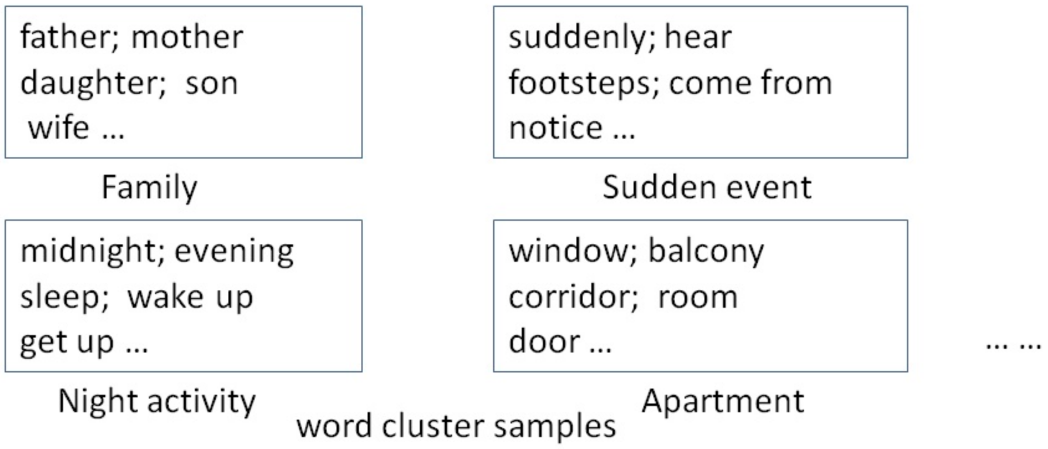


Figure 4.12: word embedding cluster

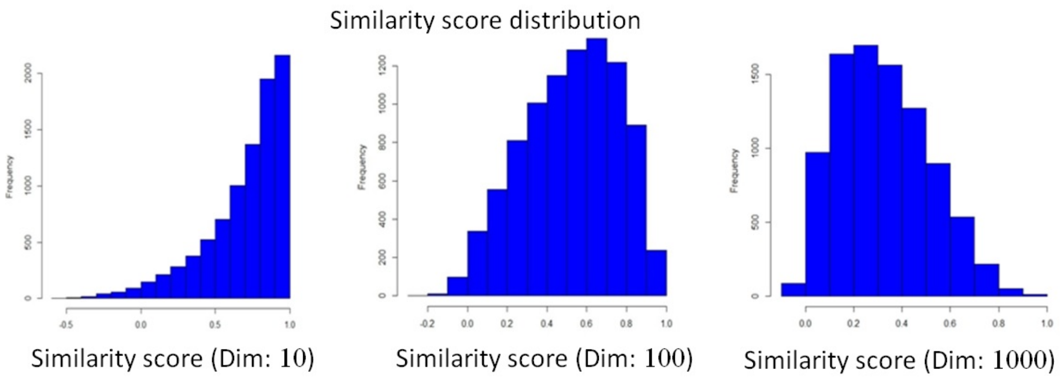


Figure 4.13: Synonymy captured by word embedding

Chapter 5

Conclusion

5.1 Summary

The focus of my study is exploring strategies that can be applied for emotion classification in minimally-supervised scenarios which is a more realistic setting in practice. In the first study, I explore the use of graph-based SSL for a document-level sentiment classification task. I exhaustively investigate the performance of label propagation in this task while varying the way to build the similarity graph which is critical to the application of graph-based SSL. I 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.

More important, I propose seed selection for graph-based SSL to alleviate the performance variance. Through the empirical evaluation on real Chinese reviews in three domains, I have confirmed that PageRank and the number of degree are effective metrics to locate the seeds with regard to stabilizing the performance. It worth emphasizing that my proposal does not require axillary language resource.

In the following study, I formulate the horror detection as a classification task, and make use of well-adopted supervised learning method (SVM) to resolve it. I recognize the feature sparsity is the cause that make conventional supervised learning approaches cannot perform well when the number of training data is very limited. In order to densify the feature space, I take measures of three unsupervised learning approaches, Brown clustering, LDA and word embedding to gain feature mappings from a large scale of unlabeled corpus. After a series of experiments, I confirm that word embedding is the optimal candidate among those three methods.

In order to change the word embedding's demerit in ignoring the task specifics, I propose a novel method to integrate supervised a-priori into the existing word embedding methods. Meanwhile, I also provide two ways to gain the supervised information. The empirical results demonstrate the effectiveness of my proposal.

5.2 Future Direction

While most study rely on bag-of-word to train the word embedding model, there is pioneer [LGRG14] to explore deep and sophisticated regularity in model training. Due to the complexity essence of language, it is necessary to exploit advanced characteristics to analyze text. Sentiment analysis is the experimental plot. Moreover, a few literature theories such as script theory [Tom78] could be helpful in the task of predicting readers perception, since it is a more potential way to intimate the readers thought. Finally, though researchers usually recognize sentiment analysis as a independent task, the other NLP problems may be benefit from it. Actually, such kind of jointly learning strategy has been proposed in [CW08].

In my future study, I will investigate the application of my proposal: task-oriented word embedding in the other classification tasks such as topic categorization to confirm its genetics. The scalability and performance measured in computing time will be my next focus, and implementation of word embedding on newly emerging platforms such as Apache Spark[1], 0xData H2O[2] and Cloudera oryx[3] will be interesting trying. I will also consider the one-class strategy, since it is not practical to judge all the potential classes in advance where we need to identify one specific type of documents from the bunch of documents.

Finally, It is will be very interesting to put emotion classification into practical usages such as product recommendation or online advertisement.

[1]https://spark.apache.org/

[2]http://0xdata.com/h2o/

[3]https://github.com/cloudera/oryx

Appendices

Appendix A

Publication List

A.1 International Journal

Yong Ren, Nobuhiro Kaji, Naoki Yoshinaga, and Masaru Kitsuregawa, Sentiment Classification in Under-Resourced Languages Using Graph-based Semi-supervised Learning Methods, IEICE Transactions on Information and Systems, Special Issue: Data Engineering and Information Management, April 2014

A.2 International Conference

Yong Ren, Nobuhiro Kaji, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa, Sentiment Classification in Resource-Scarce Languages by using Label Propagation. In Proceedings of PACLIC, pages 420-429, December 2011 (refereed)

A.3 Domestic Conference

Yong Ren, Naoki Yoshinaga, Nobuhiro Kaji, and Masaru Kitsuregawa, Detecting Horrible Episodes: Hint Fiction as a Case Study, SIG-FPAI 2013, November 2013

Yong Ren, Nobuhiro Kaji, Naoki Yoshinaga, and Masaru Kitsuregawa, Humor Identification in Microblog, iDB 2013, July 2013

Yong Ren, Nobuhiro Kaji, Naoki Yoshinaga, and Masaru Kitsuregawa, Semi-supervised sentiment Classification in Resource-Scarce Language: A Comparative Study, IEICE Tech. Rep., vol. 112, no. 172, DE2012-26, pp. 59-64, August 2012

Yong Ren, Nobuhiro Kaji, Naoki Yoshinaga, and Masaru Kitsuregawa, Mining Representative Posts in Sina Weibo, DEIM2012, March 2012

Ren Yong, Sentiment Classification in Resource-Scarce Languages by using Label Propagation, Young Researcher Association for NLP Studies (YANS), September 2011 (Poster)

Appendix B

Miscellaneous

Figure B.1 is the vocabulary distribution in the book "Moby Dick". We could see that it conforms to the power law distribution [BA99], which means that there will be a lot of less frequently appearing words and the possibility to include them in the training data is low.

Figure B.2 exhibits my demo on emotion classification on Sina Weibo, where the horizontal axis indicates the popular topics and the corresponding bars represent users attitudes on the related topics. We could gain the crowd feeling on the given target with easy.

In order to investigate the scalability of graph-based SSL algorithms, I conduct an empirical evaluation on a large scale dataset that consist of $1,046,968$ positive reviews and $13,654$ negative reviews. I should mention that this is a highly unbalanced dataset in terms of sentiment polarity. I randomly choose 200 positive reviews (balanced in sentiment polarity) as labeled seeds, and 2000 (also balanced in sentiment polarity) as test data. The remaining reviews are taken as unlabled data.

Though the number of reviews is not very large, the number of edges is 1.3 billion! I adopt the implementation of label propagation on GraphChi

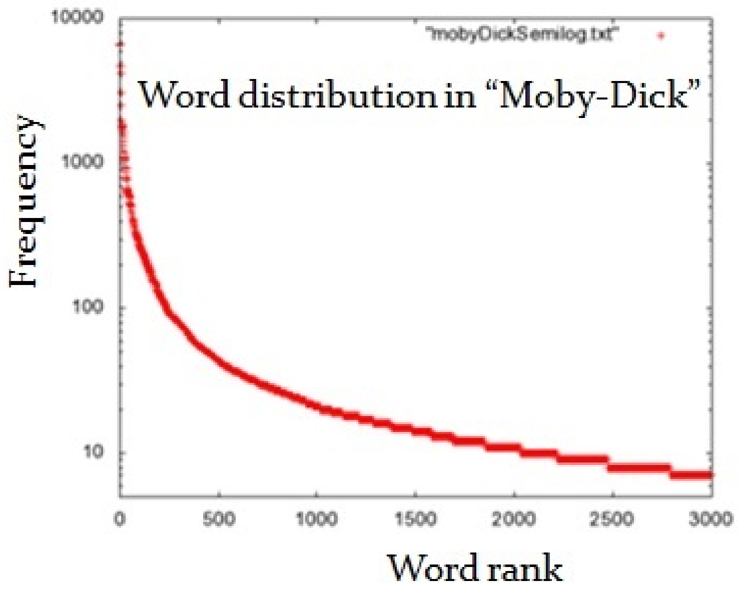


Figure B.1: Power law distribution of vocabulary in "Moby Dick"

[KBG12] during the evaluation. It takes 49 minutes to finish the classification, and all the negative test data are mis-classified as positive as there are much more positive reviews in the unlabeled data. Actually, imbalanced sentiment classification [LWZL11, WLZ+11] is another challenging field, but it is beyond the scope of my study.

It is worth pointing out that the performance bottleneck for the large scale deployment of graph-based SSL locates in the graph building phase other than the following iterative computation. Figure B.3 exhibits the time increasing with the number of vertices, and we can clearly observe that the time is increasing exponentially with the number of vertices.

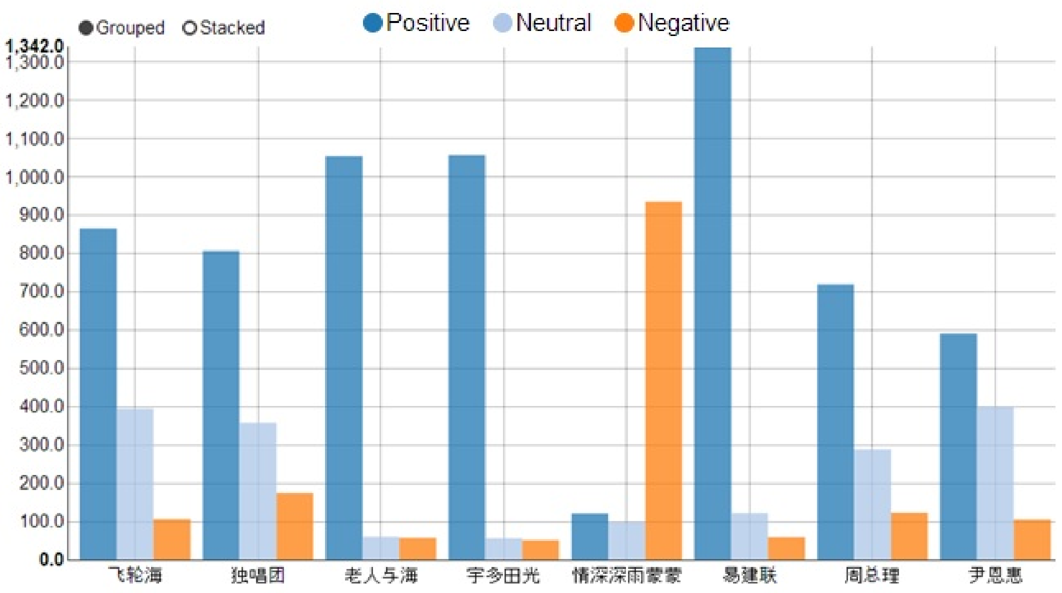


Figure B.2: My demo on emotion classification on Sina Weibo

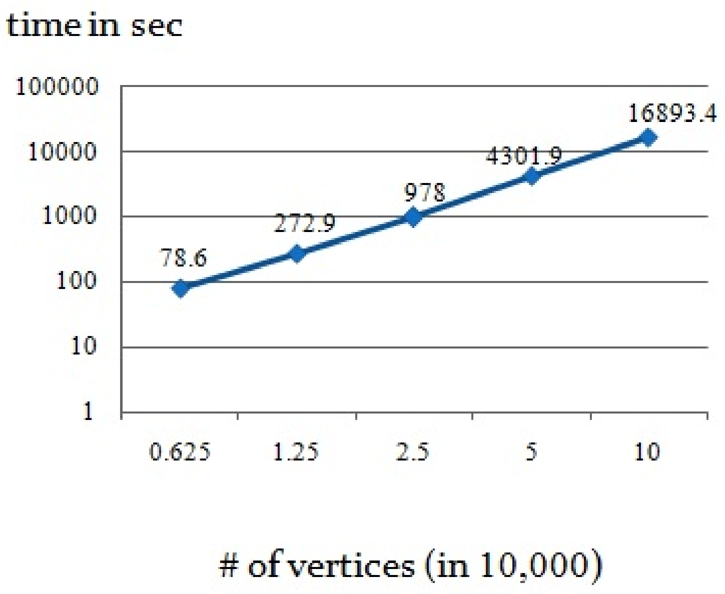


Figure B.3: Graph building time

Bibliography

[BA99] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[BC$^+$08] Y-lan Boureau, Yann L Cun, et al. Sparse feature learning for deep belief networks. In *Advances in neural information processing systems*, pages 1185–1192, 2008.

[BDM$^+$92] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

[BE10] Samuel Brody and Noemie Elhadad. An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 804–812. Association for Computational Linguistics, 2010.

[Ben09] Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[Bho09] Plaban Kumar Bhowmick. Reader perspective emotion analysis in text through ensemble based multi-label classification framework. *Computer and Information Science*, 2(4):P64, 2009.

[BM98] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, COLT' 98, pages 92–100. ACM, 1998.

[BMP06] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128. Association for Computational Linguistics, 2006.

[BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[Bra01] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(163), 2001.

[BSS$^+$06] Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2006.

[BSS$^+$08] Shumeet Baluja, Rohan Seth, D. Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceedings of the 17th international conference on World Wide Web*, pages 895–904, 2008.

[CW08] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.

[DN09] Sajib Dasgupta and Vincent Ng. 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*, pages 701–709, 2009.

[EFP75] Paul Ekman, Wallace V Friesen, and Consulting Psychologists Press. *Pictures of facial affect.* Consulting Psychologists Press, 1975.

[Gam05] Michael Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the 20th international conference on Computational Linguistics*, pages 841–847, 2005.

[GBB11] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International*

Conference on Machine Learning (ICML-11), pages 513–520, 2011.

[HBB10] Matthew Hoffman, Francis R Bach, and David M Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864, 2010.

[HKYT13] Takayuki Hasegawa, Nobuhiro Kaji, Naoki Yoshinaga, and Masashi Toyoda. Predicting and eliciting addressee's emotion in online dialogue. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 964–972, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[HOT06] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[Jac12] Paul Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 11(2):37–50, February 1912.

[Joa99] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proc. ICML*, pages 200–209, 1999.

[KBG12] Aapo Kyrola, Guy E Blelloch, and Carlos Guestrin. Graphchi: Large-scale graph computation on just a pc. In *OSDI*, volume 12, pages 31–46, 2012.

[KCC08] Terry Koo, Xavier Carreras, and Michael Collins. Simple semi-supervised dependency parsing. 2008.

[KZC+13] Suin Kim, Jianwen Zhang, Zheng Chen, Alice H Oh, and Shixia Liu. A hierarchical aspect-sentiment model for online reviews. In *AAAI*, 2013.

[LC08] Kevin Hsin-Yih Lin and Hsin-Hsi Chen. Ranking reader emotions using pairwise loss minimization and emotional distribution regression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 136–144. Association for Computational Linguistics, 2008.

[Le13] Quoc V Le. Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8595–8598. IEEE, 2013.

[LGRG14] Omer Levy, Yoav Goldberg, and Israel Ramat-Gan. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2, 2014.

[Lia05] Percy Liang. Semi-supervised learning for natural language. In *MASTER THESIS, MIT*, 2005.

[Liu12] Bing Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012.

[LL13] Igor Labutov and Hod Lipson. Re-embedding words. ACL, 2013.

[LM14] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. *CoRR*, abs/1405.4053, 2014.

[LPLN09] Honglak Lee, Peter Pham, Yan Largman, and Andrew Y Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in neural information processing systems*, pages 1096–1104, 2009.

[LW09] Dekang Lin and Xiaoyun Wu. Phrase clustering for discriminative learning. 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*, pages 1030–1038, 2009.

[LWZL11] Shoushan Li, Zhongqing Wang, Guodong Zhou, and Sophia Yat Mei Lee. Semi-supervised learning for imbalanced sentiment classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1826, 2011.

[LXWZ13] Shoushan Li, Yunxia Xue, Zhongqing Wang, and Guodong Zhou. Active learning for cross-domain sentiment classification. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 2127–2133. AAAI Press, 2013.

[LYC07] Kevin Hsin-Yih Lin, Changhua Yang, and Hsin-Hsi Chen. What emotions do news articles trigger in their readers? In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 733–734. ACM, 2007.

[MC04] Tony Mullen and Nigel Collier. Proceedings of the 14th acm international conference on information and knowledge management. In *Proc. EMNLP*, pages 412–418, 2004.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[MDP$^+$11] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 142–150. Association for Computational Linguistics, 2011.

[MGZ04] Scott Miller, Jethran Guinness, and Alex Zamanian. Name tagging with word clusters and discriminative training. In *Proceedings of HLT*, pages 337–342, 2004.

[MH08] Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2008.

[MS99] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing.* MIT Press, 1999.

[MS05] Rada Mihalcea and Carlo Strapparava. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 531–538. Association for Computational Linguistics, 2005.

[MTO05] Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. Sentiment classification using word sub-sequences and dependency sub-trees. In *Advances in Knowledge Discovery and Data Mining*, pages 301–311, 2005.

[MYZ13] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, June 2013.

[NJT05] Zheng-Yu Niu, Dong-Hong Ji, and Chew Lim Tan. Word sense disambiguation using label propagation based semi-supervised learning. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 395–402, 2005.

[PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. In *Proceedings of the 8th international conference on World Wide Web*, pages 161–172, 1999.

[PL08] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2:1–135, January 2008.

[PLV02] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86, 2002.

[PNS$^+$10] Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 751–760, New York, NY, USA, 2010. ACM.

[PY10] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[RKY$^+$11] Yong Ren, Nobuhiro Kaji, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa. Sentiment classification in resource-scarce languages by using label propagation. In *Proc. PACLIC*, pages 420–429, 2011.

[RR09] Delip Rao and Deepak Ravichandran. Proceedings of the 12th conference of the european chapter of the association for computational linguistics. In *Proc. EACL*, pages 675–682, 2009.

[SHD14] Ruhi Sarikaya, Geoffrey E Hinton, and Anoop Deoras. Application of deep belief networks for natural language understanding. *IEEE/ACM Transactions on Audio, Speech & Language Processing*, 22(4):778–784, 2014.

[SM08] Vikas Sindhwani and Prem Melville. Document-word coregularization for semi-supervised sentiment analysis. In *Proc. ICDM*, pages 1025–1030, 2008.

[SMG10] Sanasam Ranbir Singh, Hema A Murthy, and Timothy A Gonsalves. Feature selection for text classification based on gini coefficient of inequality. *Journal of Machine Learning Research-Proceedings Track*, 10:76–85, 2010.

[SPH+11] Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics, 2011.

[SSUB11] Michael Speriosu, Nikita Sudan, Sid Upadhyay, and Jason Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the First workshop on Unsupervised Learning in NLP*, pages 53–63, 2011.

[TC09] Partha Pratim Talukdar and Koby Crammer. New regularized algorithms for transductive learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 442–457, 2009.

[TC11] Yi-jie Tang and Hsin-Hsi Chen. Emotion modeling from writer/reader perspectives using a microblog dataset. *Sentiment Analysis where AI meets Psychology (SAAIP)*, page 11, 2011.

[Tom78] Silvan S Tomkins. Script theory: Differential magnification of affects. In *Nebraska symposium on motivation.* University of Nebraska Press, 1978.

[TRB10] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

[Tur02a] Peter D Turney. 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*, pages 417–424. Association for Computational Linguistics, 2002.

[Tur02b] Peter D. Turney. 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*, pages 417–424, 2002.

[Vap95] Vladimir N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., 1995.

[VBGHM10a] Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 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*, pages 777–785, 2010.

[VBGHM10b] Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 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*, pages 777–785, 2010.

[Wan09] Xiaojun Wan. 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, pages 235–243, 2009.

[WBL^{+}13] Taifeng Wang, Jiang Bian, Shusen Liu, Yuyu Zhang, and Tie-Yan Liu. Psychological advertising: exploring user psychology for click prediction in sponsored search. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 563–571. ACM, 2013.

[WLZ^{+}11] Zhongqing Wang, Shoushan Li, Guodong Zhou, Peifeng Li, and Qiaoming Zhu. Imbalanced sentiment classification with multi-strategy ensemble learning. In *Asian Language Processing (IALP), 2011 International Conference on*, pages 131–134. IEEE, 2011.

[Wu82] F. Y. Wu. The potts model. *Rev. Mod. Phys.*, 54(1):235–268, Jan 1982.

[YCS09] Tae Yano, William W Cohen, and Noah A Smith. Predicting response to political blog posts with topic models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 477–485. Association for Computational Linguistics, 2009.

[YZL09] Qiang Ye, Ziqiong Zhang, and Rob Law. Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. *Expert Systems with Applications*, 36(3):6527–6535, 2009.

[YZL10] Bing Qin Yanyan Zhao and Ting Liu. Integrating intra- and inter-document evidences for improving sentence sentiment classification. *ACTA AUTOMATICA SINICA*, 36(10):1417–1425, 2010.

[ZG02] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002.

[ZG09] Xiaojin Zhu and Andrew B. Goldberg. *Introduction to Semi-Supervised Learning.* Morgan & Claypool Publishers, 2009.