Sentiment Analysis on Social Media for Stock Market Prediction

by

Thien Hai Nguyen

submitted to
Japan Advanced Institute of Science and Technology
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Supervisor: Associate Professor Kiyoaki Shirai

School of Information Science
Japan Advanced Institute of Science and Technology

September 2015

To my family

Abstract

Sentiment analysis is important in academic as well as commercial point of views. It is a task to extract people's opinions, attitudes and emotions toward entities. There are many applications of sentiment analysis such as investigation of product reviews, opinion summarization and stock market prediction. Specifically, opinion mining in the financial domain could help to make an accurate algorithm for stock market prediction. The goals of this research are: (1) studying the aspect-based sentiment analysis to identify the polarity of an aspect term or aspect category in a sentence, (2) studying the sentiment analysis on the financial domain to predict the stock market.

We propose a new topic model, Topic Sentiment Latent Dirichlet Allocation (TSLDA), to infer the topics and their sentiments simultaneously. With the observation that the topics are usually represented by nouns, whereas the opinion words are the adjectives or adverbs, words in sentences are drawn from distributions depending on its categories: topic category, opinion category and others. In addition, different topics, which are represented by word distributions, will have different opinion word distributions. Finally, to capture the sentiment meanings such as positive, negative or neutral of the opinion words for each topic, we distinguish opinion word distributions for different sentiment meanings. TSLDA is used for aspect-based sentiment analysis as well as sentiment analysis for stock market prediction.

To identify the sentiment categories for aspect terms in the first goal, an unsupervised method "ASA w/o RE" is considered. This model calculates the sentiment value of the aspect by summing over the scores of all opinion words divided by their distances to that aspect. However, the opinion words related to the aspects will have higher affection to the sentiments of them. We firstly extract the aspect-opinion relations by using a proposed tree kernel based on the constituent and dependency trees. Then, "ASA w/o RE" is extended as "ASA with RE" by integrating these aspect-opinion relations. The experiment results show that the integration of aspect-opinion relation extraction is useful for aspect-based sentiment analysis. Next, three supervised methods RNN, AdaRNN and our proposed PhraseRNN are investigated. RNN and AdaRNN convert a dependency tree of a sentence to a binary tree. PhraseRNN combines the dependency tree and list of phrases from the constituent tree to create a phrase dependency tree. This is further converted to a target dependent binary tree. In these three models, the representation model of the aspect is constructed by recursively combining the two child nodes into a parent node in bottom-up manner. The top node is used as the representation for the aspect and fed into a logistic regression to predict the sentiment category of the aspect. The results indicate that our PhraseRNN achieved better performance than unsupervised methods "ASA w/o RE" and "ASA with RE". In addition, our PhraseRNN is better

5.35% accuracy and 7.89% F-measure than the ordinary RNN, 5.78% accuracy and 16.37% F-measure than AdaRNN. Therefore, our PhraseRNN is much effective than RNN and AdaRNN for the aspect-based sentiment analysis.

Two topic models JST and our TSLDA, are used to extract the sentiment for aspect categories in the first goal. By mapping the topics/sentiments inferred by JST or TSLDA to human topics/sentiments, the latent topics and sentiments are used to identify the aspect categories and their sentiments in each document. Our TSLDA outperforms JST model in almost all metrics in three datasets. As a result, our TSLDA is better than JST model for aspect-based sentiment analysis.

The second goal is to predict the stock price or movement using the sentiment analysis on social media. Stock price prediction is a very challenging task because the stock prices are affected by many factors. The Efficient Market Hypothesis and random walk theory said that it could not be predictable with more than about 50% accuracy. On the other hand, some researches specified that the stock market prices could be predicted at some degree. Around 56% accuracy are often reported as satisfying results.

With the assumption that integration of the sentiments from the social media can help to improve the predictive ability of models, we evaluate and compare three feature sets for stock price prediction and seven feature sets for stock movement prediction. Three employed methods to predict future stock prices are Price Only, Human Sentiment and Sentiment Classification. The first method uses only historical prices. The second combines the past prices of the stock with the sentiments annotated by posters, whereas third method uses both human and automatically classified sentiments. The results of regression models indicate that these sentiments are not useful to predict the future stock price. For the stock movement prediction, in addition to three previous models, additional four features are used: LDA-based Method, JST-based Method, TSLDA-based Method and Aspect-based Sentiment Method. Latent topics are extracted in LDA-based Method, whereas both latent topics and sentiments are exploited in the JST-based Method and TSLDA-based Method. In Aspect-based Sentiment Method, not latent but explicit topics and sentiments that appear in the sentence are incorporated into the prediction model. In addition, to address the question how automatic sentiment analysis contributes the prediction, we evaluate the automatically identified sentiment against the human annotated sentiment. The results show that our TSLDA-based and Aspect-based Sentiment Method outperform others in terms of the accuracy. The average accuracy on prediction of 5 and 18 stocks of TSLDA-based and Aspect-based Sentiment method are 56.43% and 54.41%, respectively. Besides, our method is comparable to the method using manually annotated sentiments. Therefore, the automatic sentiment analysis can be the alternative of the manual annotation. In addition, the important contribution of our experiment is that we evaluate our method for many stocks (18 stocks) and for a long time period of the test set (four months).

In future work, a nonparametric topic model for TSLDA which can guess the number of topics and sentiments by itself will be explored. In aspect term polarity identification, we will investigate the way to learn the weight parameters in "ASA with RE" method

from the training dataset to capture more accurately how the aspect-opinion relations contribute to the sentiment of the aspect. In addition, we will try to develop more sophisticated stock prediction model to also predict the degree of the change by setting more fine grained classes such as 'great up', 'little up', 'little down', 'great down' and so on.

Keywords: Text Mining, Sentiment Analysis, Opinion Mining, Stock Prediction, Social Media, Message Board, Tree Kernel, Topic Model, Recursive Neural Network, Support Vector Machine.

Acknowledgments

My studies at the **Japan Advanced Institute of Science and Technology** (JAIST) have been an incredible journey, where I have grown tremendously, both academically and personally. None of this would have been possible without the support of many people.

First of all, I would like to sincerely thank my great supervisor, **Associate Professor Kiyoaki Shirai** at JAIST, for his kindly guidance and supports. As my supervisor, he taught me a lot of things not only the knowledge about natural language processing but also the research methodology, developing the new idea, and solving problems. Without his guidance and encouragement, my work could not have been accomplished. I feel really lucky to be one of his students.

Next, I wish to send my deep acknowledgments to my terrific committee, consisting of **Associate Professor Nguyen Le Minh**, **Professor Satoshi Tojo** and **Professor Ho Tu Bao** at JAIST, and **Associate Professor Inui Takashi** at University of Tsukuba, for the time they spent reading my dissertation.

I wish to express my deep thanks to the **Doctoral Research Fellow** (DRF) Scholarship Program of JAIST for the financial supports during my research.

I would like to thank **Professor Akira Shimazu** of the School of Information Science at JAIST. He has given me a lot of valuable comments to improve my research during the seminars of the lab. He always listen to my problems and give me kind suggestion.

I would like to sincerely thank **Associate Professor Julien Velcin** of ERIC laboratory at University Lyon 2, for his guidance and support during my internship research in France.

I received a lot of help from members in **Shimazu & Shirai Laboratory** of JAIST. I owe my greats thanks to all of them

Finally, I would like to give special thanks to my family for their sacrifice, love and understanding.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

We firstly introduce the background and motivation of this research in Section 1.1. Then, the research problems are presented in Section 1.2. We describe our goals and the contributions of this dissertation. Finally, in Section 1.3, the dissertation structure is explained.

1.1 Background

Sentiment analysis (also known as **opinion mining**) is a research area in the field of text mining and natural language processing. Generally speaking, it aims to determine an attitude, opinion and emotions of people toward entities, aspects and topics in a document [1, 2]. In other words, the task of sentiment analysis is to classify the polarity (e.g: positive, negative, neutral and so on) of the document, sentence, or aspect. Applications of the sentiment analysis include tasks to determine how excited someone is about a product such as an upcoming movie, food at a restaurant, or how people think about the movement of stock price of a given company. For example, before consumers buy a product or use a service, they try to look for opinions of other users about it, whereas companies want to survey their customers' opinions to improve their products.

Stock price forecasting is very important in the trading on the stock market as well as planning of business activity [3]. For stock traders such as investors, they would predict the stock price and buy a stock before the price rises, or sell it before its value declines. An accurate prediction algorithm can help investors make high profits. On the other hand, by knowing the stock of the company will go up or down, managers of the company can control the capital and make a better future strategy decision. However, building an accurate stock prediction model is still a challenging problem. From a psychology perspective, emotion plays a significant role in the decision making process. A message, tweet, or news article may influence emotion of investors or traders, which may indirectly influence the stock price. Therefore, in addition to historical prices, the current stock market is affected by the mood of society. The overall social mood with respect to a given company might be one of the important variables which affect the stock price of that company. Nowadays, the emergence of online social networks makes available large amounts of mood data. Therefore, incorporating information from social media with the

historical prices can improve the predictive ability of models.

The rise of Twitter [1], Facebook [2], message boards, blogs, review sites and social sites has motivated people express their opinions about anything publicly and more frequently. Among the social media, stock message boards give users a place to ask questions, find information and discuss rumors regarding a chosen stock. This kind of social media creates an online community for the investors to discuss the stocks and the future prices. On the internet, Yahoo hosts on the largest and most popular online communication with boards for over 6000 stocks [3]. However, it is impossible to manually analyze the sentiments or mood in such massive text data in social media. Sentiment analysis system automatically processing a large amount of user-generated information has become a popular research topic recently.

1.2 Research Problems and Contributions

The goal of our research is to develop a model to predict the stock prices using information from social media (Message Board). In our proposed method, a model that predicts the stock value at t using features derived from information at $t-1$ and $t-2$, where t stands for a transaction date, will be trained by supervised machine learning. Apart from the mood information, the stock prices are affected by many factors such as microeconomic and macroeconomic factors. However, this research only focuses on how the mood information from social media can be used to predict the stock price. We will mainly aim at extracting the mood information by sentiment analysis on social media data. Then, these sentiments will be integrated into a model to predict stocks. To achieve this goal, discovering the topics and sentiments in a large amount of social media is very important to get the opinions of investors as well as events of companies. However, sentiment analysis on social media is very difficult. The text is usually short, contains many misspellings, uncommon grammar constructions and so on. In addition, the literature shows conflicting results in sentiment analysis for stock market prediction. Some researchers report that sentiments from social media have no predictive capabilities [4, 5], while other researchers have reported either weak or strong predictive capabilities [6]. Therefore, how to use opinions in social media for stock price predictions is still an open problem.

We concentrate on two problems as follows:

1. **Sentiment analysis on general domains**
   The purpose of this task is to build an effective sentiment analysis system which extracts the sentiment of a given aspect in a sentence. Figure 1.1 shows the architecture of this problem. We focus on two subtasks: aspect term polarity identification and aspect category polarity identification. The former tried to identify the sentiment category for a given aspect term in the sentence. The aspect terms are words

[1] https://twitter.com
[2] https://www.facebook.com
[3] http://finance.yahoo.com/mb/AAPL/

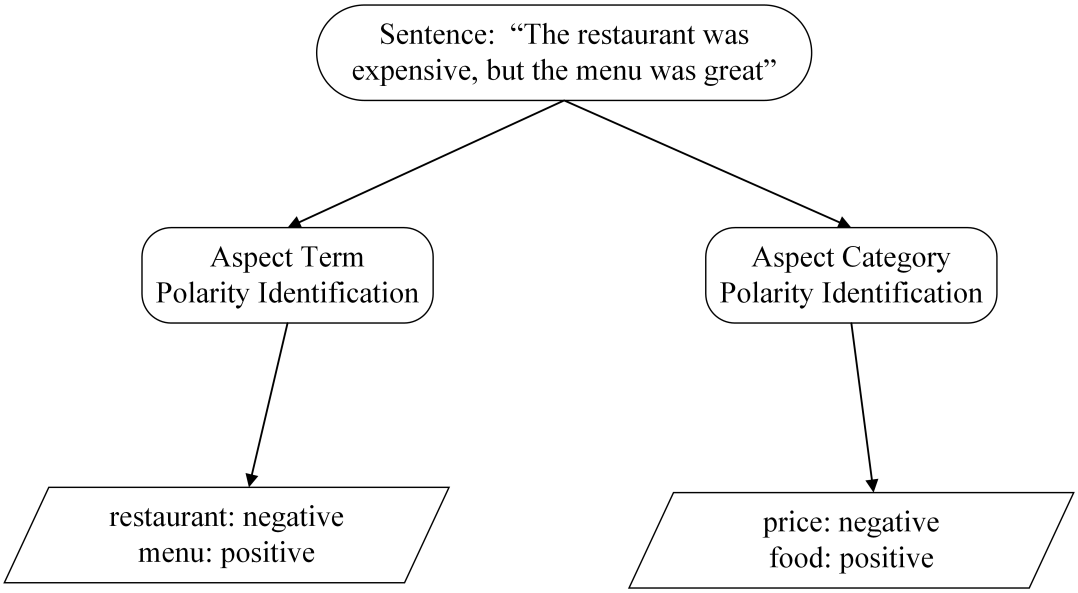

Figure 1.1: Architecture of Problem 1

or phrases appearing in the sentence, which represent the aspect, attribute or properties of an entity. On the other hand, the latter deals with the sentiments of aspect categories. The aspect category is a pre-defined aspect, attribute or property of an entity usually represented by groups of the aspect terms. Note that they do not necessarily appear in the sentence. For example, given the sentence "The restaurant was expensive, but the menu was great", the first subtask will aim at identifying the sentiment negative and positive for aspect term "restaurant" and "menu". The second subtask will aim at identifying negative and positive for aspect categories "price" and "food", where "price" and "food" are the pre-defined aspect categories of the restaurants. Our contributions for these subtasks are summarized as follows:

- We apply two existing tree kernels for aspect-opinion relation extraction. In addition, a new tree kernel based on the combination of the existing two tree kernels is proposed.
- We propose a new method "ASA with RE" (Apsect-based Sentiment Analysis with Relation Extraction) for aspect term polarity identification enhanced by the automatically identified aspect-opinion relations.
- We propose a new supervised method PhraseRNN for polarity identification based on recursive neural network. By combing the dependency tree and constituent tree, our PhraseRNN constructs the representation of the aspect based on the relationship between nodes in phrases and between phrases.
- We examine two topic model methods for aspect category polarity identification. In addition, a new way to map between hidden topics/sentiments to human topics/sentiments is proposed.

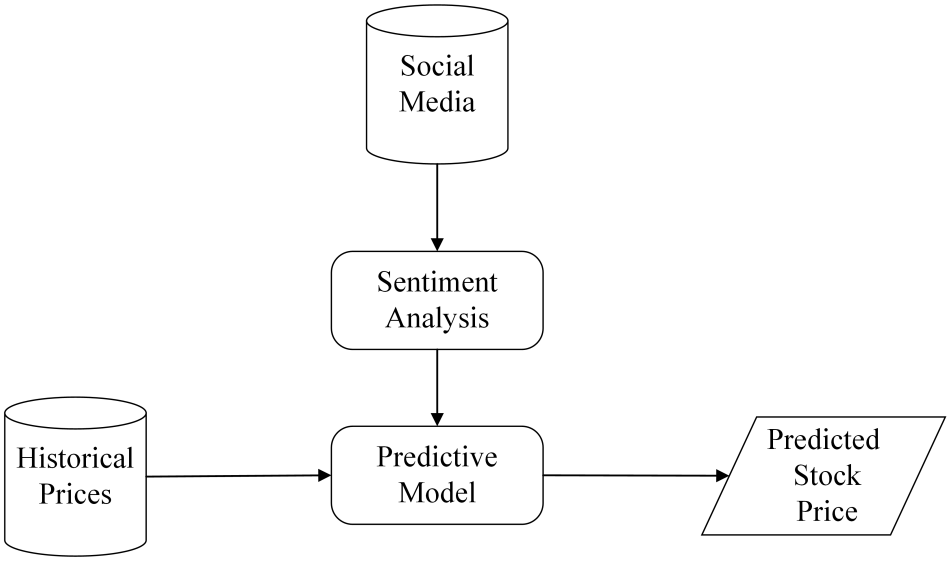

Figure 1.2: Architecture of the Proposed Framework toward Problem 2

2. **Sentiment analysis on financial domain and its application to stock prediction**
   The purpose of this task is to deploy a sentiment analysis to extract information from users in financial domain. Then, it is integrated into models to predict the future stock price. Figure 1.2 shows the architecture of the proposed framework. Our contributions for this task are summarized as follows:

   - We evaluate integration of sentiment analysis from social media into the stock price prediction and stock movement prediction. Three and seven feature sets are employed in regression models and classification model to predict the stock, respectively.
   - We propose a new feature "topic-sentiment" for the stock market prediction model. We use three methods to capture this feature: "JST-based Method", "TSLDA-based Method" and "Aspect-based Sentiment Method". The first two methods are topic models which can extract simultaneously topics and sentiment in the documents. The other is an unsupervised model.
   - A new method, "Aspect-based Sentiment Method" is proposed for stock market prediction. In contrast to JST and TSLDA, the aspects and sentiments are not hidden. They are words or phrases existing in the sentences.
   - Another contribution is large scale evaluation. Most of the research is limited on predicting for one stock, and the number of instances (transaction dates) in a test set is very low such as 14 or 15 instances [6, 7]. With only a few instances in the test set, the conclusion might be insufficient. This is the first research that shows good prediction results evaluated on a list of many stocks in a test set containing many transaction dates.

1.3 Dissertation Outline

This dissertation is structured as follows. Chapter 2 gives a brief introduction to several statistical machine learning methods including Support Vector Machine, Recursive Neural Network, Logistic Regression, Linear Regression and Support Vector Regression. Then we discuss some previous work on sentiment analysis, aspect-based sentiment analysis and sentiment analysis for stock market prediction.

Chapter 3 presents our study on topic modeling for sentiment analysis. We first describe the background for some probability distributions including dirichlet and multinomial distribution. Next, the simplest latent topic model, Latent Dirichlet Allocation, is explained. Then, a joint sentiment topic model, JST, is discussed. Finally, we introduce our topic model Topic Sentiment Latent Dirichlet Allocation (TSLDA) which can extract simultaneously the topics and sentiments in the document.

In Chapter 4, we investigate the task of aspect-based sentiment analysis which is comprised of two subtasks: aspect term polarity identification and aspect category polarity identification. For the first subtask, we first introduce an unsupervised model "ASA w/o RE" (Aspect-based Sentiment Analysis without Relation Extraction). Then, this model is further improved by integrating aspect-opinion relation extraction module. We then present how to extract aspect-opinion relations by using tree kernel based on constituent and dependency trees. Next, a supervised method PhraseRNN is introduced. For the second subtask, two topic models JST and our TSLDA are applied. Finally, the evaluations of these two subtasks are discussed.

Chapter 5 presents our study on integration of the sentiment analysis into the stock market prediction. We first describe how to collect dataset from social media. Then, we introduce three feature types for stock price prediction and seven feature types for stock movement prediction models. Then, the effectiveness of the sentiment for stock predictions is evaluated. We also show the effectiveness of our TSLDA topic model and "Aspect-based Sentiment" method for the sentiment analysis toward stock market prediction.

Figure 1.3 shows a graphical sketch of the structure of Chapter 3, 4 and 5. The texts in red indicate our main contributions in each chapter.

Finally, Chapter 6 will summarize the dissertation with our main contributions as well as the remaining problems. Some extendable parts of this study for future research directions are also presented.

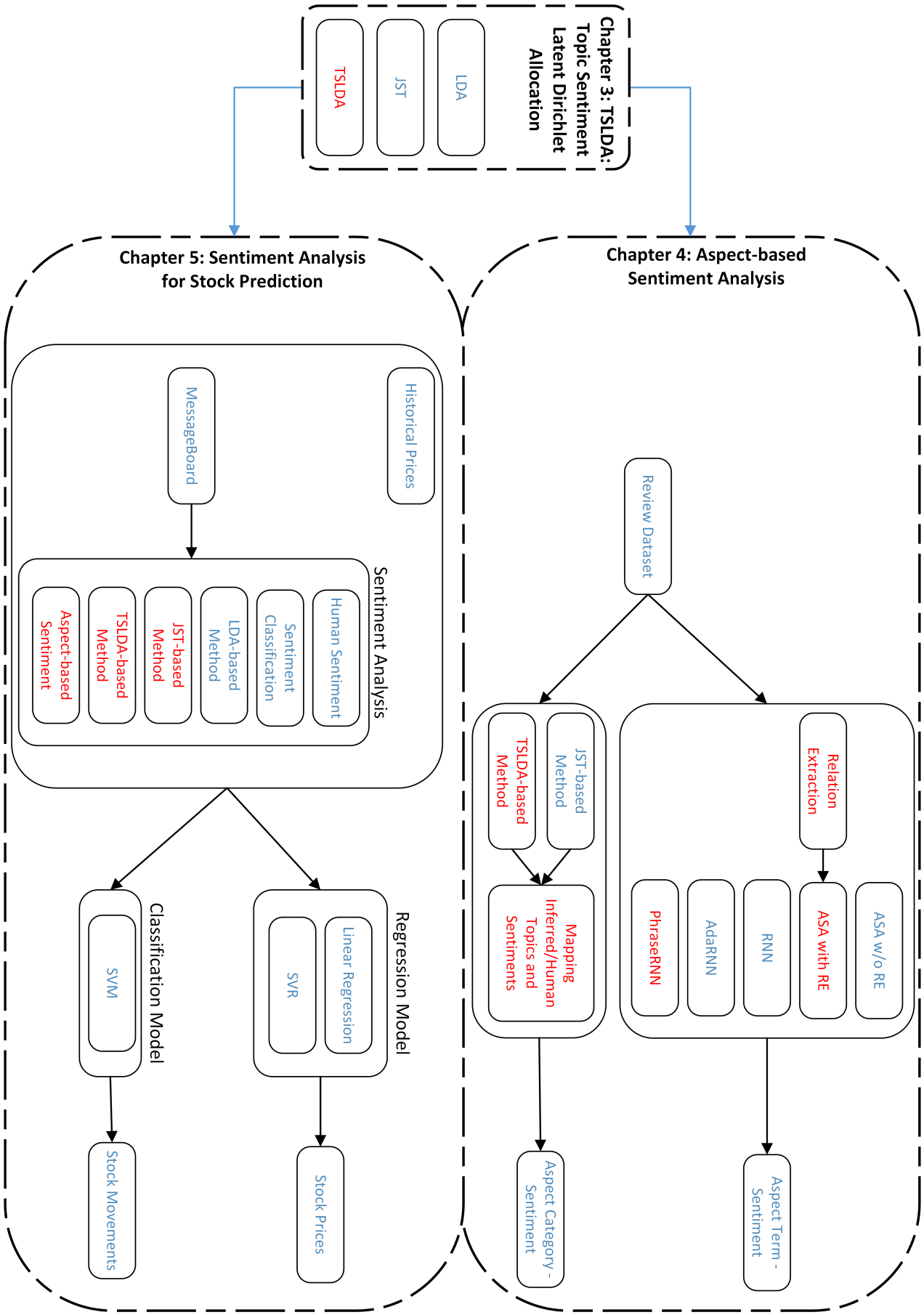

Figure 1.3: Structure of the Main Contents of this Dissertation

Chapter 2

Related Work

In this chapter, we firstly present some background knowledge about statistical machine learning models which are used in this dissertation in Section 2.1. Then, in Section 2.2, the current studies on sentiment analysis are surveyed. We discuss some related work on aspect-based sentiment analysis in Section 2.3. Finally, previous methods on sentiment analysis for stock market prediction are introduced in Section 2.4.

2.1 Statistical Machine Learning Models

We give a brief introduction to common statistical machine learning models that will be employed in this dissertation: Support Vector Machine, Recursive Neural Network, Logistic Regression, Linear Regression and Support Vector Regression.

2.1.1 Support Vector Machine

Support Vector Machine (SVM) is a statistical machine learning technique. The current standard soft margin of it was proposed by Cortes and Vapnik [8]. It is a successful modeling and prediction tool for a variety of applications in many areas such as computer vision, handwriting recognition, pattern recognition and statistical natural language processing. In the context of natural language processing, it was successfully used to text classification [9], word sense disambiguation [10], syntactic parsing [11], semantic parsing [12], sentiment analysis [13, 14], machine translation [15], information extraction [16] and so on.

Suppose that the training data is $\{(x^{(i)}, t^{(i)})\}, 1 \leq i \leq N, t^{(i)} \in \{-1, 1\}$. $t^{(i)}$ stands for the class of i-th data. A two-class SVM is a linear model of the form:

$$y(x) = w^T \Phi(x) + b \tag{2.1}$$

where $\Phi(x)$ is a feature space transformation, w and b is the parameters of the model. The new data point x is classified based on the sign of $y(x)$.

Intuitively, in binary classification task with linear cases, SVM will find a hyperplane

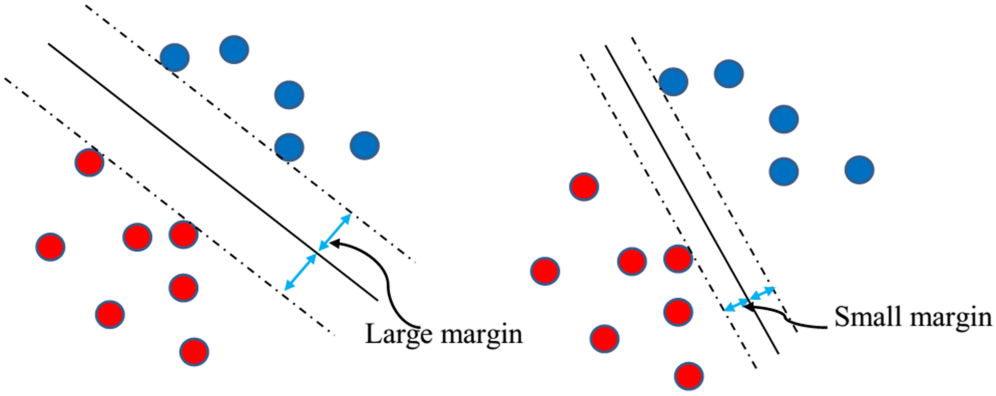

Figure 2.1: An Example of Margins in SVM

that separates positive and negative instances. The margin is defined as the perpendicular distance between the decision boundary (straight line) and the closest of the data points (points touching with the dash lines) as shown in Figure 2.1. There are many such decision boundaries. Among these possible decision boundaries, an optimal one is the hyperplane with maximum margin.

Because the training data is linearly separable in feature space, there exists at least one selection of the parameters w and b to satisfy the following two conditions:

1. For instances having $t^{(i)} = 1$: $y(x^{(i)}) > 0$
2. For instances having $t^{(i)} = -1$: $y(x^{(i)}) < 0$

For all training instances, $t^{(i)}y(x^{(i)}) > 0$. The distance of a data point $x^{(i)}$ to the decision boundary is as:

$$\frac{t^{(i)}y(x^{(i)})}{\|w\|} = \frac{t^{(i)}(w^T\Phi(x^{(i)}) + b)}{\|w\|} \tag{2.2}$$

Because margin is the distance from closest point $x^{(i)}$, the maximum margin solution is:

$$\underset{w,b}{argmax} \left\langle \frac{1}{\|w\|} \underset{i}{min}[t^{(i)}(w^T\Phi(x^{(i)}) + b)] \right\rangle \tag{2.3}$$

For the classification with $K > 2$ classes, there are two common approaches. The first one is *one-versus-the-rest*, the other is *pairwise*. The former method constructs K separate SVMs. The k^{th} SVM is trained with instances from class k as positive and the remaining as negative instances. The prediction for a new input x is a class having highest $y_k(x)$. The latter method trains a binary SVM for all $\frac{K(K-1)}{2}$ possible pairs of the classes. The class having the highest number of votes is selected as the prediction class for an instance.

2.1.2 Recursive Neural Network

A recursive neural network (RNN) is a type of deep neural network. In natural language processing, it has been proven to be successful in learning sequence as well as tree

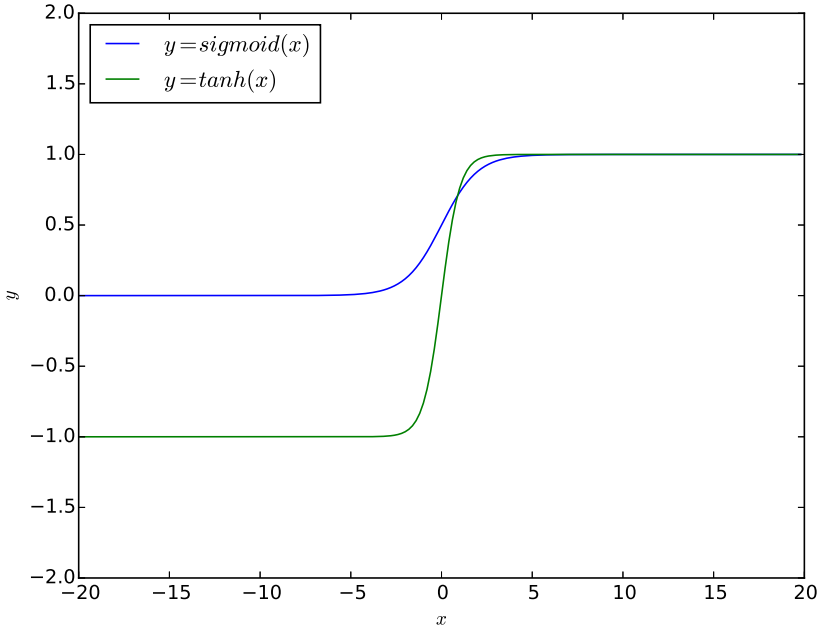

Figure 2.2: Tanh and Sigmoid Functions

structures [17, 18].

RNN is created by applying a set of weights over a structure such as a sequence or tree. In particular, child nodes are combined into parent nodes by using a weight matrix and a nonlinear function f such as tanh (2.4) or sigmoid (2.5). Figure 2.2 shows the tanh and sigmoid functions.

$$tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{2.4}$$

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{2.5}$$

Figure 2.3 shows an example of a simple architecture of RNN. Input nodes a, b, c, $\in \Re^d$ are d-dimensional vectors, while $W \in \Re^{d \times 2d}$ is the parameters to be learnt. The parent nodes p_1 and p_2 must have the same dimension with the input nodes. They could be used recursively as inputs to the next composition. The parent nodes p_1 is created from two child nodes b, c. Then, parent p_2 is created from a and p_1 as in Equation (2.6).

$$p_1 = f(W \begin{bmatrix} b \\ c \end{bmatrix}), \qquad p_2 = f(W \begin{bmatrix} a \\ p_1 \end{bmatrix}) \tag{2.6}$$

Similar to other neural network models, the parameters are optimized by minimizing a cost function. This can be achieved by using a local message passing in which information is sent alternately forwards and backwards through the structure. It is known as error backpropagation, or sometimes simply as backprop.

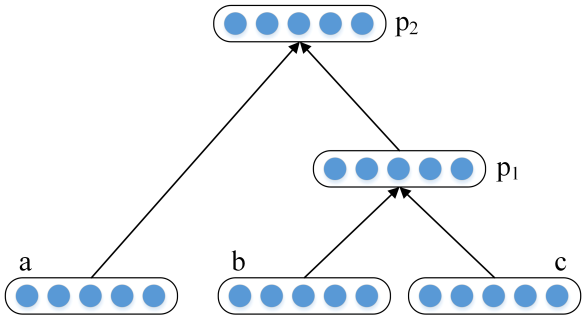

Figure 2.3: A Simple Architecture of Recursive Neural Network

2.1.3 Logistic Regression

A logistic regression is a supervised learning algorithm which can be used in many problems in natural language processing such as text classification [19, 20]. It is a probabilistic model to measure the relationship between the categorical dependent variable and one or more independent variables [21, 22]. It estimates the posterior probabilities of K classes using a logistic function. These posterior probabilities are between $[0, 1]$, and summation of them is 1. The probability of output taking the class k given the input $x^{(i)} \in \Re^n$ and parameters $\theta_1, \cdots, \theta_K \in \Re^n$ is obtained as in Equation (2.7).

$$P(y^{(i)} = k|x^{(i)}, \theta) = \frac{e^{\theta_k^T x^{(i)}}}{\sum_{j=1}^{K} e^{\theta_j^T x^{(i)}}} \tag{2.7}$$

To learn the parameter θ, logistic regressions minimize the following cost function:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{K} 1\{y^{(i)} = j\} log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{K} e^{\theta_l^T x^{(i)}}}\right] \tag{2.8}$$

Unfortunately, there is no known closed-form way to estimate the parameters that minimize the cost function. An iterative algorithm such as gradient descent is used. The iterative algorithm estimates the partial derivative of the cost function which is equal to:

$$\Delta_{\theta_j} = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}(1\{y^{(i)} = j\} - P(y^{(i)}|x^{(i)}, \theta))\right] \tag{2.9}$$

The value of θ_j is updated at each iteration:

$$\theta_j = \theta_j - \alpha\Delta_{\theta_j} \tag{2.10}$$

where α is the learning rate. Notice that the training time of Logistic Regressions is significantly more comparing to Naive Bayes, because it uses an iterative algorithm to

estimate the parameters of the model [23].

After we have estimated the parameters θ, for a new input x, the hypothesis function will need to estimate the probability $P(y = k|x, \theta)$ for each of the k possible classes. Therefore, the hypothesis function will return a k dimensional vector with the estimated probabilities:

$$h_\theta(x) = \begin{bmatrix} P(y = 1|x, \theta) \\ P(y = 2|x, \theta) \\ \cdots \\ P(y = K|x, \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{K} e^{\theta_j^T x}} \begin{bmatrix} e^{\theta_1^T x} \\ e^{\theta_2^T x} \\ \cdots \\ e^{\theta_K^T x} \end{bmatrix} \tag{2.11}$$

To predict the class of an unknown instance, the class label having the highest posterior probability is selected.

$$\hat{y} = \underset{k \in K}{argmax}\, P(y = k|x, \theta) \tag{2.12}$$

2.1.4 Linear Regression

A linear regression model is a method for modeling the relationship between a dependent variable y and one or more independent variable $x_1, \cdots, x_D$ [24]. The model takes the form as in Equation (2.13).

$$y = w_0 + w_1 x_1 + \cdots + w_D x_D + \epsilon = XW + \epsilon \tag{2.13}$$

where $w_0, \cdots, w_D$ are coefficients, W is the matrix of the coefficients, X is the matrix of the inputs. A dependent variable y is a linear relationship between independent variables with an error variable ϵ. If the number of independent variable is one, the model is called simple linear regression.

Figure 2.4 shows an example of a simple linear regression model with one independent variable x and a dependent variable y. The blue dots represent the data points. The red line is the regression line which models the linear relation between x and y.

The sum of squared residuals is defined as in Equation (2.14):

$$SSR = \sum_{i=1}^{N} \{y^{(i)} - t^{(i)}\}^2 \tag{2.14}$$

To estimate the parameters in linear regression, the simplest and most common method is ordinary least squares. Ordinary least squares minimize the sum of squared residuals. It leads to a closed form expression for the estimated value of the parameters W as in Equation (2.15).

$$\hat{W} = (X^T X)^{-1} X^T y \tag{2.15}$$

2.1.5 Support Vector Regression

Support Vector Regression (SVR) is a version of SVM for regression problems. This model was proposed by Drucker et al. [25]. By using an alternative loss function, SVM

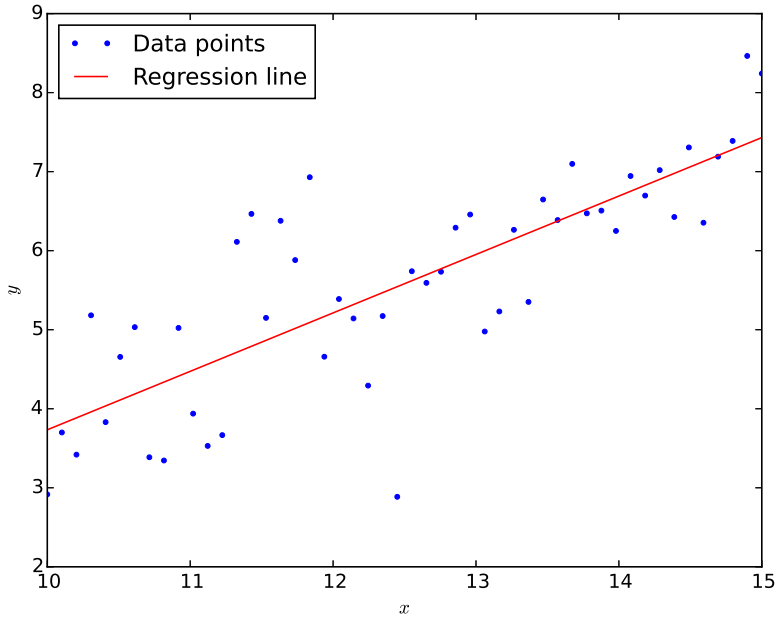

Figure 2.4: An Example of Linear Regression

can be applied to regression problems [21, 22]. This loss function must be revised to include the distance measure. An ϵ insensitive error function increases error linearly with distance beyond the insensitive region as defined in Equation (2.16). Figure 2.5 shows the ϵ insensitive error function (in red color) compared with the quadratic error function (in green).

$$E_\epsilon(y^{(i)} - t^{(i)}) = \begin{cases} 0 & \text{if } |^{(i)} - t^{(i)} | < \epsilon \\ | \, y^{(i)} - t^{(i)} \, | - \epsilon & \text{otherwise} \end{cases} \tag{2.16}$$

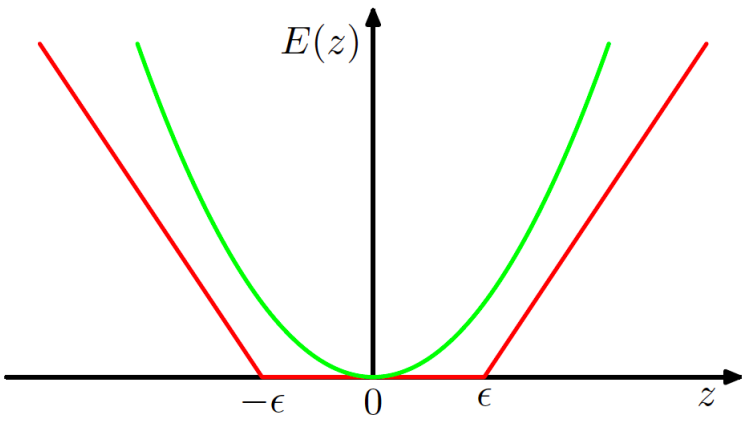

Figure 2.5: ϵ Insensitive Error Function

The objective is to minimize the regularized error function:

$$C\sum_{i=1}^{N} E_\epsilon(y(x^{(i)}) - t^{(i)}) + \frac{\lambda}{2}\|w\|^2 \tag{2.17}$$

where $y(x)$ is given by Equation (2.1) and C is an inverse regularization parameter.

We used the two slack variables $\xi^{(i)} \geqslant 0$ and $\hat{\xi^{(i)}} \geqslant 0$ for each data point $x^{(i)}$. $\xi^{(i)} > 0$ and $\hat{\xi^{(i)}} > 0$ correspond to points $t^{(i)} > y(x^{(i)}) + \epsilon$ and $t^{(i)} > y(x^{(i)}) - \epsilon$, respectively. The error function in Equation (2.17) can be rewritten as:

$$C\sum_{i=1}^{N} \xi^{(i)} + \hat{\xi^{(i)}} + \frac{\lambda}{2}\|w\|^2 \tag{2.18}$$

The error function in Equation (2.18) can be solved by introducing Lagrange multipliers and optimizing Lagrangian.

2.2 Sentiment Analysis

Let examine a review segment on iPhone quoted below to introduce the problem.

"I bought an iPhone a few days ago (1). It was such a nice phone (2). The touch screen was really cool (3). The voice quality was clear too (4). However, my mother was mad with me as I did not tell her before I bought it (5). She also thought the phone was too expensive, and wanted me to return it to the shop (6)."

There are some opinions expressed in this review segment. Sentence (1) shows a fact with no opinions or emotions in it. In sentence (2), (3) and (4), positive opinions are expressed, whereas sentence (5) and (6) are negative emotions. In addition, we noticed that all of these opinions have their targets. In sentence (2), (3) and (4), the targets are "phone", "touch screen" and "voice quality", respectively. The target of sentence (5) is "me", not related to "phone". Finally, opinion in sentence (6) expresses towards the target "price". Then, we can also see that the holders of opinions exist in these sentences. The holder of the opinions in sentences (2), (3) and (4) is the reviewer. In contrast, in sentence (5) and (6), it is the mother of the reviewer.

In general, an opinion is defined as a quintuple $(e_j, a_{jk}, so_{ijkl}, h_i, t_l)$, where e_j is a target entity, a_{jk} is an aspect/feature of the entity e_j, so_{ijkl} is the sentiment value of the opinion from the opinion holder h_i on aspect a_{jk} of entity e_j at time t_l, h_i is an opinion holder, and t_l is the time when the opinion is expressed. The importance of this definition is that we can transform the unstructured text to structured data by extracting these five components.

For instance, a user **User1** posted on Facebook on **04-20-2015** with the message: "I bought an iPhone last week. Love it so much. The design is beautiful. The touch screen is really cool. However, the phone runs out of battery quite fast." From this message, the tuples extracted by sentiment analysis module should be as follows:

- (iPhone, GENERAL, +, User1, 04-20-2015)
- (iPhone, design, +, User1, 04-20-2015)
- (iPhone, touch_screen, +, User1, 04-20-2015)
- (iPhone, battery, -, User1, 04-20-2015)

Given a set of document D, to extract the five components of opinions, we need to accomplish the following tasks:

- **Task 1** (Entity Extraction and Grouping): extract all target entities in all of documents. Then they are grouped into similar meaning groups. Each of these groups represents a unique target entity e_j.
- **Task 2** (Aspect Extraction and Grouping): extract all aspects of entities and group them into similar clusters. Each of these clusters represents a unique aspect/feature a_{jk} of the entity e_j.
- **Task 3** (Opinion Holder and Time Extraction): extract all the holders of opinions and the time when opinions are expressed.
- **Task 4** (Aspect Sentiment Identification): identify if the opinion so_{ijkl} of the opinion holder h_i on aspect a_{jk} of entity e_j at time t_l is positive, negative or neutral.

Among these tasks, task 2 and task 4 attract a lot of researches because of their difficulty. In this dissertation, we also focus on these two tasks.

Sentiment analysis usually analyzes the opinions on various levels: document, sentence or aspect. The most common approach is to determine the sentiment at the document level. The polarity of the whole document will be determined. JPang et al. used naive bayesian classification and support vector machine to classify each movie review into two classes: positive and negative [26]. They concluded that using bag-of-words as features performed well with either naive bayes or SVM. Pak and Paroubek used n-gram as feature representations in naive bayes, SVM and CRF for twitter sentiment classification [27]. Moraes et al. presented an empirical comparison between SVMs and Artificial Neural Networks regarding document-level sentiment analysis [28]. A deep belief network was proposed for sentiment classification in online reviews [29]. Instead of using a machine learning, some researches have proposed custom methods for sentiment classification such as score functions based on positive and negative words [30]. Ohana and Tierney used SentiWordNet to count positive and negative term scores to determine sentiment orientation [31]. Paltoglou and Thelwall enhanced the accuracy of the classification by using feature weighting schemes [32].

The second approach is sentence level sentiment analysis, that is, a finer-grained analysis of a given document. It tries to classify each sentence into one of some predefined sentiment categories. At the sentence level, the same techniques used at document level can also be applied to individual sentences [33]. Most of researches assumed the sentence

expresses only a single opinion. However, a single sentence may contain multiple opinions [34]. Therefore, it is useful to analyze compound and complex sentences.

The most detailed approach is aspect level sentiment analysis. It extracts the sentiment expressed toward an aspect or feature in a given sentence. The next section will discuss some previous work about it.

The development of lexical resources for sentiment analysis attracted a lot of attention. Most of the work relies on some lexical resources for opinion mining [35, 14, 36]. General Inquirer consists of 1915 positive terms and 2291 negative terms [37]. HM consists of 657 positive and 679 negative adjectives [38]. Opinion Finder lexicon is a list of 8221 English words in positive and negative categories [39]. A lexicon with affective norms for English words, ANEWS, is released [40]. AFINN is an English word list with 2477 words constructed for sentiment analysis of Twitter messages [41]. SentiStrength estimates the strength of positive and negative sentiments [42]. NRC tagged English words with emotion ratings [43]. Finally, SentiWordNet assigns to each synset of WordNet [44] three sentiment scores: positivity, negativity and objectivity [45].

Instead of using a general polarity word resource such as SentiiWordNet, other research tried to combine different lexical resources to a unified lexical resource [36]. Other developed a domain specific polarity word resource such as laptop, financial domains by adjusting the general lexical resources or building the list from a thesaurus such as WordNet [46, 47, 48, 49].

2.3 Aspect-based Sentiment Analysis

Aspect-based sentiment analysis has been found to play a significant role in many applications such as opinion mining on product reviews or restaurant reviews. The popular approach is to define a sentiment score of a given aspect by the weighted sum of opinion scores of all words in the sentence, where the weight is defined by the distance from the aspect [1, 2].

To extract the aspects, many techniques such as supervised learning, semi-supervised learning, topic modeling and clustering are used. Conditional Random Fields [50, 51] and Hidden Markov Models [52, 53, 54] are employed in previous research. Su et al. presented a method to extract the implicit aspects by using a clustering method with mutual reinforcement [55].

Popescu and Etzioni extracted the explicit and implicit features for product reviews and mined the polarities of these features [56]. By using the web as a corpus to identify the features, their model achieved 22% higher precision and 3% lower recall compared to the model proposed by Hu and Liu [48]. Jakob and Gurevych used Conditional Random Fields to extract opinion targets in a sentence [50]. Tokens, POSs, short dependency paths, word distances and opinion words are used as features for sequence labeling the tokens in the sentence.

Using syntactic relations between aspects and opinion words, Qiu et al. simultaneously extracted both aspect and opinion [57, 58]. Their methods used a bootstrapping process

to extract aspects from opinion words and extracted aspects. Opinion words are extracted from extracted aspects, given and extracted opinion words.

Socher et al. proposed a Recursive Neural Tensor Network to predict sentiment labels for all phrases in the sentences [59]. They concluded that their model can capture negation and its scope at various tree levels.

Dong et al. proposed AdaRNN model for target-dependent Twitter sentiment classification [60, 61]. Given a sentence, different target words have different binary trees converted from a dependency tree of the sentence. In addition, they employed multi-compositional functions and adaptively selected them depending on the linguistic tags and the combined vectors.

Kim and Hovey proposed a method for identifying the topics and expressed opinions given a sentence in online news [62]. By exploiting semantic structures from FrameNet [63], they extracted the topics of the opinion words based on the labeled semantic roles.

Brody and Elhadad presented an unsupervised system for extracting aspects and determining sentiments in the review text [64]. They used a topic model to automatically infer the aspects. To identify the sentiment, they proposed a method for automatically deriving an unsupervised seed set of positive and negative adjectives.

There are some researches trying to use topic models to extract both the topic and sentiment for some domains such as online product, restaurant and movie review dataset. ASUM is a model for extracting both the aspect and sentiment for online product review dataset [65]. Joint sentiment/topic model (JST) is another model to detect the sentiment and topic simultaneously, which was applied for movie review dataset [66]. These models assume that each word is generated from a joint topic and sentiment distribution. It means that these models do not distinguish the topic word and opinion word distributions.

Besides the general opinion words, topic models considering aspect-specific opinion words were also proposed. MaxEnt-LDA hybrid model can jointly discover both aspects and aspect-specific opinion words on a restaurant review dataset [67], while FACTS, CFACTS, FACTS-R and CFACTS-R model were proposed for sentiment analysis on a product review data [68]. However, one of the weaknesses of these methods is that there is only one opinion word distribution corresponding to one topic (aspect). It makes difficult to know which sentiment (e.g. positive or negative) is expressed by the opinion words on that topic.

To overcome this drawback, we propose a new topic model called Topic Sentiment Latent Dirichlet Allocation (TSLDA), which estimates different opinion word distributions for individual sentiment categories for each topic. To the best of our knowledge, such a model has not been proposed. TSLDA is suitable for not only sentiment analysis for stock prediction but also general sentiment analysis of the document, sentence and aspect.

2.4 Sentiment Analysis for Stock Prediction

Stock market prediction is one of the most attracted topics in academic as well as real life business. Many researches have tried to address the question whether the stock market

can be predicted. Some of the researches were based on the random walk theory and the Efficient Market Hypothesis (EMH). According to the EMH [69, 70], the current stock market fully reflects all available information. Hence, price changes are merely due to new information or news. Because news in nature happens randomly and is unknowable in the present, stock prices should follow a random walk pattern and the best bet for the next price is the current price. Therefore, they are not predictable with more than about 50% accuracy [71]. On the other hand, various researches specify that the stock market prices do not follow a random walk, and can be predicted at some degree [6, 72, 7]. Degrees of accuracy at 56% hit rate in the predictions are often reported as satisfying results for stock predictions [73, 74, 75].

Besides the efficient market hypothesis and the random walk theories, there are two distinct trading philosophies for stock market prediction: fundamental analysis and technical analysis. The fundamental analysis studies the financial conditions and operations of the company as well as macroeconomic indicators to predict stock price. On the other hand, the technical analysis depends on historical and time-series prices. Price moves in trends, and history tends to repeat itself. Some researches have tried to use only historical prices to predict the stock price [76, 77]. To discover the pattern in the data, they used bayesian network [76, 77], time-series method such as Auto Regressive model, Moving Average model, Auto Regressive Moving Average model [76] and so on.

As mentioned in Section 2.2, sentiment analysis has been found to play a significant role in many applications such as product and restaurant reviews [1, 2]. There are some researches trying to apply sentiment analysis as an information source to improve the stock prediction model. There are two main textual sources from which researchers have incorporated information into financial models. In the past, the main source was the news [78, 73], and in recent years, social media sources. A simple approach is combining the textual content with the historical prices through the linear regression model.

Most of the previous work primarily used the bag-of-words as text representation that are incorporated into the prediction model. Luss and d'Aspremont modeled price movements of financial assets by using bag-of-words to categorize press releases [79]. Schumaker and Chen tried to use different textual representations such as bag-of-words, noun phrases and named entities for financial news [73]. Then this information was integrated with linear regression and support vector machine regression as predictive models. They applied their models to estimate a discrete stock price 20 minutes after a news article was released. The results show 0.04261 Mean Square Error, 57.1% directional accuracy and 2.06% return in a simulated trading engine. However, the textual representation is just the words or named entity tags, not exploiting so much about the mood information.

Samangooei et al. used tweets of the stock AAPL to predict its prices in 2010 [80]. Instead of using TF-IDF as a feature weighting for each word, they proposed a new weighting scheme called DF-IDF. The bag-of-words with the proposed term weighting is used in an ordinary linear regression as the prediction model. Their model achieved 12.53 mean square error. However, their method for sentiment analysis is quite simple and they only experimented with only one stock.

Zhang and Fuehres used naive bayes to classify the messages from message boards into three classes: buy, hold and sell [4]. The number of relevant messages in these three classes was aggregated into a single measure of bullishness. They investigated three aggregation functions as a number of alternatives to bullishness. They were integrated into the regression model. However, they concluded that their model does not successfully predict stock returns.

Zhang et al. measured collective hope and fear on each day and analyzed the correlation between these indices and the stock market indicators [81]. They used the mood words to tag each tweet as fear, worry, hope and so on. They concluded that the emotional tweet percentage significantly negatively correlated with Down Jones, NASDAQ and S&P 500, but had significant positive correlation to VIX. However, they did not use their model to predict the stock price values.

Bollen et al. analyzed the text content of daily Twitter by two mood tracking tools: OpinionFinder and Google Profile of Mood States [6, 82]. The former measures positive and negative mood. The latter measures mood in terms of six dimensions (Calm, Alert, Sure, Vital, Kind and Happy). They used the Self Organizing Fuzzy Neural Network model to predict DJIA values. The results show 86.7% direction accuracy (up or down), Mean Absolute Percentage Error 1.79%. However, their test period is very short (from December 1 to December 19, 2008). Even though, they achieved a high accuracy, there are only 15 transaction dates in their test set. With such a short period, it might not be sufficient to conclude the effectiveness of their method.

Xie et al. proposed a novel tree representation based on semantic frame parsers [83]. They indicated that this representation performed significantly better than bag-of-words. By using stock prices from Yahoo Finance, they annotated all the news labels in a transaction date as going up or down categories. However, the weakness of this assumption is that all the news in one day will have the same category. In addition, this becomes a document classification problem, not stock prediction.

Rechenthin et al. used Yahoo Finance Message Board to incorporate into the stock movement prediction [84]. They tried to use various classification models to predict stock. They used the explicit sentiments and predicted sentiments obtained by a classification model with the bag-of-words and meta-features. However, their research does not focus on sentiment analysis on these message boards.

Vu et al. integrated sentiment features in Twitter for stock prediction [7]. They detect the sentiment of tweets as positive, neutral and negative by using a keyword-based algorithm. Their model achieved around 75% accuracy. However, their test period is very short, from 8^{th} to 26^{th} in September, 2012 which contains only 14 transaction dates.

Si et al. developed a nonparametric topic model for Twitter messages to predict the stock market [74]. They proposed a continuous Dirichlet Process Mixture (cDPM) model to learn the daily topic set. Then, a sentiment time series was built based on these topics. The advantage of this method is that the model estimates the number of topics inherent in the data. However, the time period of their dataset is quite short, only three months.

Most of the research tried to extract only the opinions or sentiments of the documents

(posted message, tweets and so on). However, one important missing thing is that opinions or sentiments are expressed on topics and aspects. Therefore, understanding on which topics or aspects of a given stock people express the opinion is very important. To the best of our knowledge, there is no previous research that tries to extract simultaneously topics and sentiments for stock market prediction.

Most of the research tried to predict only one stock [6, 72, 74] and test on a short period of time [6, 7]. To the best of our knowledge, there is no research showing a good prediction result on a list of many stocks and on a long time period. We conduct experiments to predict the prices of 18 stocks over a period of one year.

Chapter 3

TSLDA: Topic Sentiment Latent Dirichlet Allocation

In this chapter, we will describe our proposed topic model, Topic Sentiment Latent Dirichlet Allocation (TSLDA), which can simultaneously extract the topics and their sentiments in the documents. We firstly introduce some background on probability distributions, including dirichlet and multinomial distributions in Section 3.1. Then, previous work on topic models will be explained. We discuss the simplest topic model, Latent Dirichlet Allocation (LDA), in Section 3.2. A previous topic model on extracting sentiments and topics, Joint Sentiment/Topic (JST), is introduced in Section 3.3. Finally, Section 3.4 explains our TSLDA model.

3.1 Probability Distributions

3.1.1 Dirichlet Distribution

Dirichlet distribution, denoted $Dirichlet(\alpha)$, is a family of continuous multivariate probability distributions with a parameter vector α of real positive values [85, 86, 21]. It can be seen as a random distribution on a finite set. In Bayesian statistics, it is very often used as the prior distribution because the posterior is also a Dirichlet distribution. Dirichlet distribution is the conjugate prior of the multinomial distribution.

The Dirichlet distribution of a K-dimensional vector x with a parameter vector $\alpha = [\alpha_1, \cdots, \alpha_K]$ has a probability density function as in Equation (3.1):

$$Dirichlet(x, \alpha) = \frac{1}{\Delta(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \tag{3.1}$$

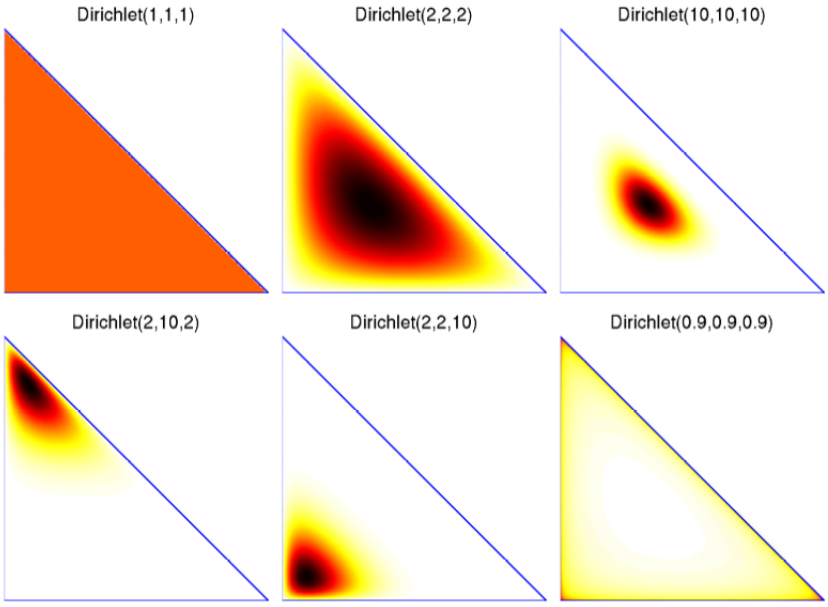

Figure 3.1: Example of Dirichlet Distributions

where $\Delta(\alpha)$ is defined as:

$$\Delta(\alpha) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{K} \alpha_i)} \tag{3.2}$$

In Equation (3.2), $\Gamma(n)$ is the gamma function for a positive integer n:

$$\Gamma(n) = (n-1)! \tag{3.3}$$

Let $\alpha_0 = \sum_{i=1}^{K} \alpha_i$. It is typically referred as the concentration parameter. It controls how the distribution is concentrated around its expected value [87]. The expectation E and variance Var of x_i are:

$$E[x_i] = \frac{\alpha_i}{\alpha_0} \tag{3.4}$$

$$Var[x_i] = \frac{\alpha_i(\alpha_0 - \alpha_i)}{\alpha_0^2(\alpha_0 + 1)} \tag{3.5}$$

Figure 3.1 shows examples of Dirichlet distributions over the 2-simplex [1] with different values of parameter vectors α. A special case of dirichlet distribution is symmetric α, where all elements of α have the same value. This symmetric dirichlet distribution is often used when no prior information favoring one component over another. When $\alpha = 1$, the symmetric dirichlet distribution is equivalent to a uniform distribution as shown in the top left of Figure 3.1.

[1] A simplex is a generalization of the triangle to arbitrary dimension. In other words, K-simplex has K + 1 vertices.

The dirichlet distribution is also a member of the exponential family. Equation (3.1) can be rewritten as:

$$Dirichlet(x, \alpha) = \frac{1}{\Delta(\alpha)} exp\left\{\sum_{i=1}^{K}(\alpha_i - 1)log(x_i)\right\} \tag{3.6}$$

3.1.2 Multinomial Distribution

Multinomial distribution, denoted $Multinomial(n, p)$ or $Multinomial(p)$ if $n = 1$, is a generalization of the binomial distribution [85, 86, 21]. A multinomial distribution has the following properties. For n independent trials, each leads to one of K categories. Each category has a given fixed probability. The multinomial distribution gives the probability of any particular success combination for these n trials.

The probability function of the multinomial distribution is shown in Equation (3.7):

$$P(x_1, \cdots, x_K | n, p_1, \cdots, p_K) = \frac{n!}{x_1! \cdots x_K!} p_1^{x_1} \cdots p_K^{x_K} \tag{3.7}$$

where $x_1, \cdots, x_K$ are the number of trials leading to categories $1, \cdots, K$, respectively. $p_1, \cdots, p_K$ are the probabilities for each category.

The expectation E and variance Var of the number of times the category i was observed over n trials are defined as in Equation (3.8) and (3.9).

$$E[x_i] = np_i \tag{3.8}$$

$$Var[x_i] = np_i(1 - p_i) \tag{3.9}$$

For example, we toss a dice three times and record the outcome on each toss. Each trial can result in 6 possible outcomes (six faces in the dice). The probability of each outcome is 1/6. The trials are independent. In particular, a particular outcome from one trial does not affect the outcome of other trials. The probability of tossing 2 outcome number 5 and 1 outcome number 6 is:

$$P(x_5 = 1, x_6 = 2 | 3, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}) = \frac{3!}{1!2!} \frac{1}{6} \left(\frac{1}{6}\right)^2 = 0.01389 \tag{3.10}$$

3.2 LDA

In natural language processing, Latent Dirichlet Allocation (LDA) is a generative probabilistic model of a corpus. It is an example of a topic model and was first presented as a graphical model for topic discovery by Blei et al [88]. LDA allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. This is similar to prob-

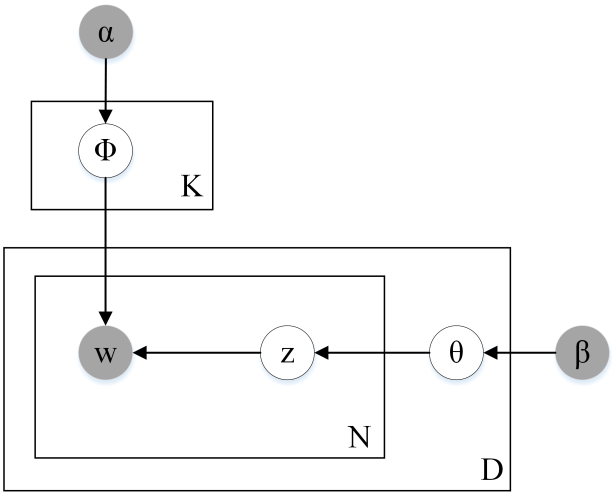

Figure 3.2: Graphical Model Representation of LDA

Table 3.1: Notations in LDA

Notation	**Definition**
α, β	hyperparameters
Φ	the distribution over words
K	the number of topics
θ	the message specific topic distribution
z	a topic
w	a word in the message d
N	the number of words in the message d
D	the number of messages

abilistic latent semantic analysis (pLSA) [89] except that the topic distribution in LDA is assumed to have a Dirichlet prior. Under a uniform Dirichlet prior distribution, pLSA model is equivalent to LDA. Figure 3.2 shows the graphical model representation of LDA. Notations in Figure 3.2 are shown in Table 3.1.

For example, a LDA model might have latent topics that can be classified as "arts" topic and "education" topic. A topic has high probabilities of generating various words, such as 'film', 'show' and 'music', which can be classified and interpreted by the viewer as "arts" topic. Naturally, the word 'film' itself will have high probability for this topic. The "education" topic also has high probabilities of generating the words such as 'school', 'student' and 'education'. Words that are not related to any topics, such as function words, will have roughly even probability between classes (or can be placed into a separate category). A topic is not explicitly defined but identified on the basis of their likelihood of co-occurrence. A word may occur in several topics with different probabilities, however, with a different typical set of neighboring words in each topic. Figure 3.3 shows an example of four topics and a document with words highly associated with these four topics. Each color in the document represents the topic in which that word was generated.

In LDA, the generative process is as follows:

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Figure 3.3: Example of Topic and Distribution of Topic in a Document in LDA

1. For each topic k:
   - Choose a distribution of topic words $\Phi_k \sim Dirichlet(\alpha)$
2. For each document d:
   - Choose a topic distribution $\theta_d \sim Dirichlet(\beta)$
   - For each word $w_{d,m}$ in the document:
     - Choose a topic assignment $z_{d,m} \sim Multinomial(\theta_d)$
     - Choose a word $w_{d,m} \sim Multinomial(\Phi_{z_{d,m}})$

From the graphical representation in Figure 3.2, we can write the joint distribution of all known and hidden variables as in Equation (3.11).

$$P(\boldsymbol{z}, \boldsymbol{w}, \theta, \varphi) = \prod_{k=1}^{K} P(\Phi_k|\alpha) \prod_{d=1}^{D} P(\theta_d|\beta) \prod_{m=1}^{N} P(z_{d,m}|\theta_d) P(w_{d,m}|\Phi_{z_{d,m}}) \tag{3.11}$$

There are various techniques to infer in LDA. The original paper used a variational Bayes approximation of the posterior distribution [88]. Gibbs Sampling [90] and expectation propagation [91] are proposed as alternatives. Gibbs sampling is a special case of Markov-Chain Monte Carlo and seems to be the best approach to approximate the posterior distribution.

The algorithm of Gibbs sampling is as follows. Let V be the vocabulary size. $N_{d,r}^{k}$ be the number of word tokens in the document d with the same word symbol r assigned to the topic k. If any of these dimensions is not limited to a specific value, we used an asterisk $*$ to denote it. For example, $N_{d,*}^{k}$ is the number of word tokens in the document d assigned to the topic k.

$-(d, m)$ stands for exclusion of the value at the m^{th} word in the document d. For example, $\boldsymbol{z}_{\boldsymbol{-(d,m)}}$ denotes all of topic assignment variables $\boldsymbol{z}$ but $z_{d,m}$.

We used square brackets for specifying the value at the index of a vector or distribution. For instance, $\alpha[v]$ denotes the value of α at index v.

Gibbs sampling will sequentially sample hidden variable $z_{d,m}$ from the distribution over these variables given the current values of all other hidden and observed variables. This distribution is estimated as in Equation (3.12).

$$P(z_{d,m} = a|\boldsymbol{z}_{\boldsymbol{-(d,m)}}, \boldsymbol{w}) \propto (N_{d,*}^{a-(d,m)} + \beta[a]) \frac{N_{*,v}^{a-(d,m)} + \alpha[v]}{\sum_{r=1}^{V} N_{*,r}^{a-(d,m)} + \alpha[r]} \tag{3.12}$$

The distribution of Φ and θ can be estimated as:

$$\Phi_k[r] = \frac{N_{*,r}^{k} + \alpha[r]}{\sum_{v=1}^{V} N_{*,v}^{k} + \alpha[v]} \tag{3.13}$$

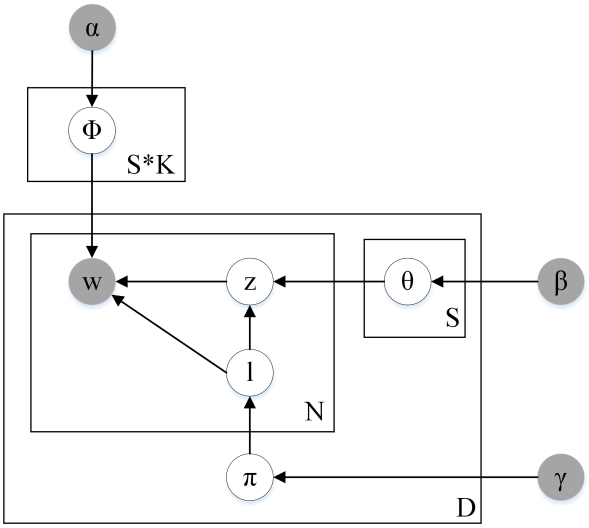

Figure 3.4: Graphical Model Representation of JST

$$\theta_d[i] = \frac{N^i_{d,*} + \beta[i]}{\sum_{k=1}^{K} N^k_{d,*} + \beta[k]} \tag{3.14}$$

3.3 JST

Joint Sentiment/Topic (JST) is an extended model of LDA and proposed by Lin and He [66]. JST considers each message as a mixture of hidden topics and sentiments. It is used to extract topics and sentiments simultaneously. Figure 3.4 shows the graphical model representation of JST. Notations in Figure 3.4 are shown in Table 3.2. In the existing framework of LDA has three hierarchical layers, where topics are associated with documents and words are associated with topics. JST models the sentiment of the document by adding an additional sentiment layer between the document and the topic layer. In LDA model, there is only one document specific topic distribution for each individual document. In contrast, each document in JST is associated with S sentiment labels. Each of sentiment labels is associated with a document specific topic distribution with the same number of the topics. A word in the document is drawn from the distribution over words defined by the topic and sentiment label.

In JST, the generative process is as follows:

1. For each topic t, sentiment s:
   - Choose a joint distribution of sentiment and topic words $\Phi_{s,t} \sim Dirichlet(\alpha)$
2. For each document d:
   - Choose a sentiment distribution $\pi_d \sim Dirichlet(\gamma)$
   - For each sentiment label l:

Table 3.2: Notations in JST

Notation	Definition
α, β, γ	hyperparameters
Φ	the distribution over words
K	the number of topics
S	the number of sentiments
θ	the message and sentiment specific topic distribution
z	a topic
w	a word in the message d
l	a sentiment label
π	the message specific sentiment distribution
N	the number of words in the message d
D	the number of messages

  - Choose a topic distribution $\theta_{d,l} \sim Dirichlet(\beta)$:

- For each word $w_{d,m}$ in the document:
  - Choose a sentiment assignment $l_{d,m} \sim Multinomial(\pi_d)$
  - Choose a topic assignment $z_{d,m} \sim Multinomial(\theta_{d,l_{d,m}})$
  - Choose a word $w_{d,m} \sim Multinomial(l_{d,m}, \Phi_{z_{d,m}})$

The joint probability of the topic assignments, sentiment assignments and words is shown in Equation (3.15):

$$P(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{l}) = P(\boldsymbol{w}|\boldsymbol{z}, \boldsymbol{l})P(\boldsymbol{z}, \boldsymbol{l}) = P(\boldsymbol{w}|\boldsymbol{z}, \boldsymbol{l})P(\boldsymbol{z}|\boldsymbol{l}, d)P(\boldsymbol{l}|d) \tag{3.15}$$

Gibbs sampling can also be applied to infer JST. Let V be the vocabulary size. $N_{j,k,d}$ be the number of times a word in the document d has been associated with the topic j and the sentiment label k. $N_{k,d}$ is the number of times the sentiment label k has been assigned to some word tokens in the document d. Each variable of interest, $z_{d,m}$ and $l_{d,m}$, are sampled from a conditional distribution as in Equation (3.16).

$$\begin{aligned} &P(z_{d,m} = a, l_{d,m} = b|\boldsymbol{z_{-(d,m)}}, \boldsymbol{l_{-(d,m)}}, \boldsymbol{w}) \\ &\propto \frac{\{N_{w_{d,m},j,k}\}_{-(d,m)} + \alpha}{\{N_{j,k}\}_{-(d,m)} + V\alpha} \times \frac{\{N_{j,k,d}\}_{-(d,m)} + \beta}{\{N_{k,d}\}_{-(d,m)} + K\beta} \times \frac{\{N_{k,d}\}_{-(d,m)} + \gamma}{\{N_d\}_{-(d,m)} + S\gamma} \end{aligned} \tag{3.16}$$

The distributions of the words in the topics and the sentiment labels are estimated as:

$$\Phi_{i,j,k} = \frac{N_{i,j,k} + \alpha}{N_{j,k} + V\alpha} \tag{3.17}$$

The distributions over topics for sentiment labels are estimated as:

$$\theta_{j,k,d} = \frac{N_{j,k,d} + \beta}{N_{k,d} + K\beta} \tag{3.18}$$

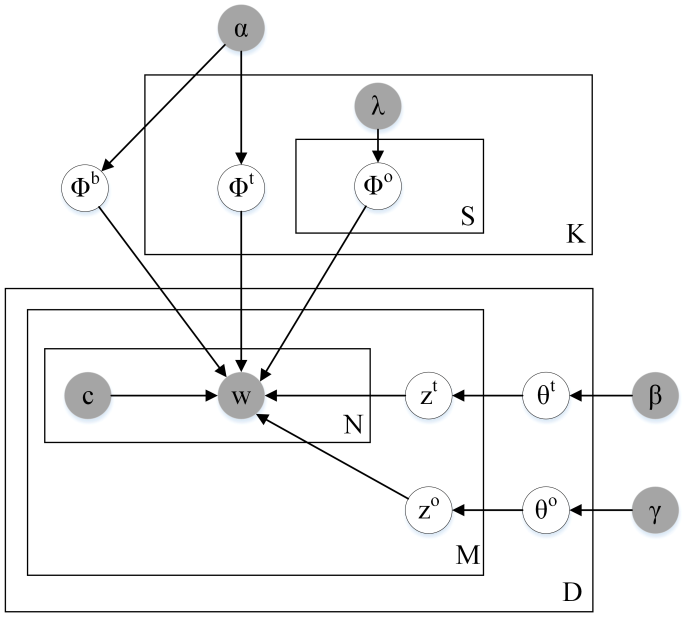

Figure 3.5: Graphical Model Representation of TSLDA

The distributions over sentiment labels for documents are:

$$\pi_{k,d} = \frac{N_{k,d} + \gamma}{N_d + S\gamma} \tag{3.19}$$

3.4 TSLDA

The proposed model TSLDA infers the topics and their sentiments simultaneously. It is an extended model of LDA [88]. We assume that one sentence expresses only one topic and one opinion on that topic. The topics are usually nouns, whereas the opinion words are the adjectives or adverbs. In TSLDA, the words in the document are classified into three categories, the topic word (category $c = 1$), opinion word ($c = 2$) and others ($c = 0$). Then, we suppose the different opinion words are used for the different topics. Depending on the topic, an opinion word may express different sentiment meaning. For example, the opinion word "low" in "low cost" and "low salary" have opposite polarity. In our model, different topics, which are also represented by word distributions, will have different opinion word distributions. Finally, to capture the sentiment meanings such as positive, negative or neutral of the opinion words for each topic, we distinguish opinion word distributions for different sentiment meanings.

Figure 3.5 shows the graphical model representation of TSLDA. Observed and hidden variables are indicated by shaded and clear circles, respectively. Table 3.3 shows the notations in Figure 3.5. The generation process in TSLDA is as follows:

1. Choose a distribution of background words $\Phi^b \sim Dirichlet(\alpha)$

2. For each topic k:

Table 3.3: Notations in TSLDA

Notation	**Definition**
α, β, γ, λ	Dirichlet prior vectors
K	the number of topics
S	the number of sentiments
Φ^b	distribution over background words
Φ^t	distribution over topic words
Φ^o	distribution over sentiment words
D	the number of documents
M_d	the number of sentences in document d
$N_{d,m}$	the number of words in sentence m in document d
θ_d^t	topic distribution for document d
θ_d^o	sentiment distribution for document d
$z_{d,m}^t$	topic assignment for sentence m in document d
$z_{d,m}^o$	sentiment assignment for sentence m in document d
$w_{d,m,n}$	n^{th} word in sentence m in document d
$c_{d,m,n}$	n^{th} word's category (background, topic or sentiment) in sentence m in document d

- Choose a distribution of topic words $\Phi_k^t \sim Dirichlet(\alpha)$
- For each sentiment s of topic k:
  - Choose a distribution of sentiment words $\Phi_{k,s}^o \sim Dirichlet(\lambda)$

3. For each document d:
   - Choose a topic distribution $\theta_d^t \sim Dirichlet(\beta)$
   - Choose a sentiment distribution $\theta_d^o \sim Dirichlet(\gamma)$
   - For each sentence m:
     - Choose a topic assignment $z_{d,m}^t \sim Multinomial(\theta_d^t)$
     - Choose a sentiment assignment $z_{d,m}^o \sim Multinomial(\theta_d^o)$
     - For each word in the sentence:
       * Choose a word $w_{d,m,n}$ as in Equation (3.20).

$$w_{d,m,n} \sim \begin{cases} Multinomial(\Phi^b) & \text{if } c_{d,m,n} = 0 \\ Multinomial(\Phi^t_{z^t_{d,m}}) & \text{if } c_{d,m,n} = 1 \\ Multinomial(\Phi^o_{z^t_{d,m}, z^o_{d,m}}) & \text{if } c_{d,m,n} = 2 \end{cases} \tag{3.20}$$

The joint probability of the topic assignments, sentiment assignments, words and categories is defined as in Equation (3.22):

$$\begin{aligned}
&P(\boldsymbol{z^t}, \boldsymbol{z^o}, \boldsymbol{w}, \boldsymbol{c}) \\
&= \int_{\boldsymbol{\theta^t}} \int_{\boldsymbol{\theta^o}} \prod_{d=1}^{D} P(\theta_d^t|\beta) P(\theta_d^o|\gamma) \prod_{m=1}^{M} P(z_{d,m}^t|\theta_d^t) P(z_{d,m}^o|\theta_d^o) d\theta^t d\theta^o \\
&\times \int_{\Phi^b} \int_{\boldsymbol{\Phi^t}} \int_{\boldsymbol{\Phi^o}} P(\Phi^b|\alpha) \prod_{k=1}^{K} P(\Phi_k^t|\alpha) \prod_{s=1}^{S} P(\Phi_{k,s}^o|\lambda) \\
&\times \prod_{d=1}^{D} \prod_{m=1}^{M} \prod_{n=1}^{N} P(w_{d,m,n}|c_{d,m,n}, z_{d,m}^t, z_{d,m}^o, \Phi^b, \boldsymbol{\Phi^t}, \boldsymbol{\Phi^o}) d\Phi^b d\boldsymbol{\Phi^t} d\boldsymbol{\Phi^o} \qquad (3.21) \\
&\propto \prod_{k=1}^{K} \Gamma(Z_d^{k,*} + \beta_k) \prod_{s=1}^{S} \Gamma(Z_d^{*,s} + \gamma_s) \\
&\times \frac{\prod_{v=1}^{V} \Gamma(W_{*,*,v,0}^{*,*} + \alpha[v])}{\Gamma(\sum_{v=1}^{V} W_{*,*,v,0}^{*,*} + \alpha[v])} \prod_{k=1}^{K} \frac{\prod_{v=1}^{V} \Gamma(W_{*,*,v,1}^{k,*} + \alpha[v])}{\Gamma(\sum_{v=1}^{V} W_{*,*,v,1}^{k,*} + \alpha[v])} \prod_{s=1}^{S} \frac{\prod_{v=1}^{V} \Gamma(W_{*,*,v,2}^{k,s} + \lambda[v])}{\Gamma(\sum_{v=1}^{V} W_{*,*,v,2}^{k,s} + \lambda[v])} \qquad (3.22)
\end{aligned}$$

Next, the way to estimate the parameters in TSLDA will be explained. First, some notations are defined as follows. $W_{d,m,v,c}^{k,s}$ is the number of times the word v with the category c appears in the sentence m in the document d, where m discusses the topic k

and the sentiment s. Let $Z_d^{k,s}$ be the number of times the document d has the topic k and the sentiment s. If any of these dimensions is not limited to a specific value, we used an asterisk $*$ to denote it. For example, $W_{*,*,v,c}^{k,s}$ is the number of appearance of combination (v, c, k, s) in any sentences in any documents. Similarly, $Z_d^{k,*}$ is the number of times the document d has the topic k with any sentiments.

A variable in bold denotes the list of the variables. For instance, $\boldsymbol{z^t}$ and $\boldsymbol{w}$ denote all topic assignments and words in all documents, respectively.

$-(d, m)$ stands for exclusion of the value in the sentence m in the document d. For example, $\boldsymbol{z^t_{-(d,m)}}$ denotes all of topic assignment variables $\boldsymbol{z^t}$ but $z_{d,m}^t$. Similarly, $Z_d^{a,*-(d,m)}$ denotes the value of $Z_d^{a,*}$ not counting times at the sentence m in the document d.

We used square brackets for specifying the value at the index of a vector or distribution. For instance, $\alpha[v]$ denotes the value of α at index v.

As in the LDA, the exact parameter estimation of TSLDA is intractable. Collapsed Gibbs Sampling was implemented for approximate inference in TSLDA. It will sequentially sample hidden variables $z_{d,m}^t$ and $z_{d,m}^o$ from the distribution over these variables given the current values of all other hidden and observed variables. In other words, in order to perform Collapsed Gibbs Sampling, conditional probability $P(z_{d,m}^t = a, z_{d,m}^o = b|\boldsymbol{z^t_{-(d,m)}}, \boldsymbol{z^o_{-(d,m)}}, \boldsymbol{w}, \boldsymbol{c})$ is calculated by marginalizing out random variables Φ^b, $\boldsymbol{\Phi^t}$, $\boldsymbol{\Phi^o}$, $\boldsymbol{\theta^t}$ and $\boldsymbol{\theta^o}$. It is defined as in Equation (3.24). Let $V_{d,m}$ be a set of words in the sentence m in the document d. V is a set of all of the words in all documents.

$$
\begin{aligned}
&P(z_{d,m}^t = a, z_{d,m}^o = b|\boldsymbol{z^t_{-(d,m)}}, \boldsymbol{z^o_{-(d,m)}}, \boldsymbol{w}, \boldsymbol{c}) \\
&= \frac{P(\boldsymbol{z^t}, \boldsymbol{z^o}, \boldsymbol{w}, \boldsymbol{c})}{P(z^t_{-(d,m)}, z^o_{-(d,m)}, \boldsymbol{w}, \boldsymbol{c})} && (3.23) \\
&\propto P(\boldsymbol{z^t}, \boldsymbol{z^o}, \boldsymbol{w}, \boldsymbol{c}) && (3.24)
\end{aligned}
$$

Combining the Equation (3.22) and (3.24), we obtained the conditional probability as

in Equation (3.25).

$$
\begin{aligned}
&P(z^t_{d,m} = a, z^o_{d,m} = b | \boldsymbol{z^t_{-(d,m)}}, \boldsymbol{z^o_{-(d,m)}}, \boldsymbol{w}, \boldsymbol{c},) \\
&\propto \prod_{k=1}^{K} \Gamma(Z_d^{k,*-(d,m)} + \beta_k) \prod_{s=1}^{S} \Gamma(Z_d^{*,s-(d,m)} + \gamma_s) \\
&\quad (Z_d^{a,*-(d,m)} + \beta_a)(Z_d^{*,b-(d,m)} + \gamma_b) \\
&\quad \prod_{k=1}^{K} \frac{\prod_{v=1}^{V} \Gamma(W^{k,*-(d,m)}_{*,*,v,1} + \alpha[v])}{\Gamma(\sum_{v=1}^{V} W^{k,*-(d,m)}_{*,*,v,1} + \alpha[v])} \prod_{s=1}^{S} \frac{\prod_{v=1}^{V} \Gamma(W^{k,s-(d,m)}_{*,*,v,2} + \lambda[v])}{\Gamma(\sum_{v=1}^{V} W^{k,s-(d,m)}_{*,*,v,2} + \lambda[v])} \\
&\quad \frac{\prod_{v=1}^{V} \prod_{j=1}^{W^{*,*}_{d,m,v,1}} (W^{a,*-(d,m)}_{*,*,v,1} + \alpha[v] + j - 1)}{\prod_{j=1}^{W^{*,*}_{d,m,*,1}} (\sum_{v=1}^{V} W^{a,*-(d,m)}_{*,*,v,1} + \alpha[v] + j - 1)} \\
&\quad \frac{\prod_{v=1}^{V} \prod_{j=1}^{W^{*,*}_{d,m,v,2}} (W^{a,b-(d,m)}_{*,*,v,2} + \lambda[v] + j - 1)}{\prod_{j=1}^{W^{*,*}_{d,m,*,2}} (\sum_{v=1}^{V} W^{a,b-(d,m)}_{*,*,v,2} + \lambda[v] + j - 1)}
\end{aligned}
\tag{3.25}
$$

By removing the constants which looping over words not in the sentence m in the document d, we simplified the conditional probability to the final derivative as in Equation (3.26).

$$
\begin{aligned}
&P(z^t_{d,m} = a, z^o_{d,m} = b | \boldsymbol{z^t_{-(d,m)}}, \boldsymbol{z^o_{-(d,m)}}, \boldsymbol{w}, \boldsymbol{c},) \\
&\propto (Z_d^{a,*-(d,m)} + \beta[a])(Z_d^{*,b-(d,m)} + \gamma[b]) \\
&\quad \times \frac{\prod_{v=1}^{V_{d,m}} \prod_{j=1}^{W^{*,*}_{d,m,v,1}} (W^{a,*-(d,m)}_{*,*,v,1} + \alpha[v] + j - 1)}{\prod_{j=1}^{W^{*,*}_{d,m,*,1}} (\sum_{v=1}^{V} W^{a,*-(d,m)}_{*,*,v,1} + \alpha[v] + j - 1)} \\
&\quad \times \frac{\prod_{v=1}^{V_{d,m}} \prod_{j=1}^{W^{*,*}_{d,m,v,2}} (W^{a,b-(d,m)}_{*,*,v,2} + \lambda[v] + j - 1)}{\prod_{j=1}^{W^{*,*}_{d,m,*,2}} (\sum_{v=1}^{V} W^{a,b-(d,m)}_{*,*,v,2} + \lambda[v] + j - 1)}
\end{aligned}
\tag{3.26}
$$

Finally, samples obtained from Collapsed Gibbs Sampling can be used to approximate the multinomial parameter sets. The distributions of topics and sentiments in the document d are estimated as in Equation (3.27) and (3.28).

$$\theta_d^t[a] = \frac{Z_d^{a,*} + \beta[a]}{\sum_{k=1}^{K} Z_d^{k,*} + \beta[k]} \tag{3.27}$$

$$\theta_d^o[b] = \frac{Z_d^{*,b} + \gamma[b]}{\sum_{s=1}^{S} Z_d^{*,s} + \gamma[s]} \tag{3.28}$$

The background word distribution, topic word distribution of the topic k and sentiment word distribution of the sentiment s for k are estimated in Equation (3.29), (3.30) and (3.31), respectively.

$$\Phi^b[r] = \frac{W_{*,*,r,0}^{*,*} + \alpha[r]}{\sum_{v=1}^{V} W_{*,*,v,0}^{*,*} + \alpha[v]} \tag{3.29}$$

$$\Phi_k^t[r] = \frac{W_{*,*,v,1}^{k,*} + \alpha[r]}{\sum_{v=1}^{V} W_{*,*,v,1}^{k,*} + \alpha[v]} \tag{3.30}$$

$$\Phi_{k,s}^o[r] = \frac{W_{*,*,v,2}^{k,s} + \lambda[r]}{\sum_{v=1}^{V} W_{*,*,v,2}^{k,s} + \lambda[v]} \tag{3.31}$$

The remained problem is how to determine the categories c (topic, opinion or other) of the words in the documents, since they are usually not observed in the dataset. One of the solution is using a machine learning method to automatically identify the category of each word. In this dissertation, a rule-based algorithm is applied. Consecutive nouns are considered as topic words. If a word is not a noun and in a list of opinion words in SentiWordNet [45], it is considered as an opinion word. The rest of words are classified as background words.

Chapter 4

Aspect-based Sentiment Analysis

This chapter focuses on aspect-based sentiment analysis, which is a task to identify the sentiment expressed towards aspects of entities. Models to predict the sentiment categories (positive, neutral or negative) of the given aspect in the sentence are introduced. As described in Section 1.2, there are two kinds of tasks of aspect-based sentiment analysis: aspect term polarity identification and aspect category polarity identification. The former focuses on aspect terms which present in the sentence. On the other hand, the latter focuses on aspect categories that is pre-defined coarse-grained aspects of a certain type of a product or service. They are represented as groups of aspect terms and do not necessarily appear in the sentence.

4.1 Methods for Aspect Term Polarity Identification

Aspect term polarity identification is a task to identify the sentiment categories for a given aspect term in a sentence. In this section, we tried to integrate the relation extraction model to aspect-based sentiment analysis. Intuitively, not all opinion words in the sentence represent emotion on the given aspect. The opinion words related to the given aspect will have more influence on the sentiment of that aspect. Our method firstly identifies the aspect-opinion relations in the sentence by tree kernel method. Then, it calculates the sentiment score for each aspect in the sentence considering these extracted relations. We will show that our model is effective and improve the model without aspect-opinion relation extraction.

4.1.1 Aspect-based Sentiment Analysis without Relation Extraction: a baseline model

We used a popular algorithm for calculating a score of a given aspect [1, 2]. Even though this algorithm is simple, it can perform well in many cases. Given a sentence, which contains a set of aspects $A = \{a_1, \cdots, a_m\}$ and a set of opinion words $OW = \{ow_1, \cdots, ow_n\}$, the sentiment score for each aspect a_i is calculated as in Equation (4.1). The opinion word

that appears near the aspect would deeply influence the polarity of it. Therefore, the sentiment value of the aspect is defined as the summation over all opinion values divided by their distances to that aspect. The aspect is categorized as positive, negative and neutral if $sentimentValue(a_i)$ is greater than 0.25, less than -0.25 and other.

$$sentimentValue(a_i) = \sum_{j=1}^{|OW|} \frac{opinionValue(ow_j)}{distance(a_i, ow_j)} \tag{4.1}$$

Opinion words were identified based on SentiWordNet [45] that is a lexical resource for opinion mining. Three sentiment scores (positivity, objectivity and negativity) are assigned to each word in SentiWordNet. Opinion value $opinionValue(ow)$ of a opinion word ow is defined as Equation (4.2).

$$opinionValue(ow) = \frac{positivityScore - negativityScore}{positivityScore + negativityScore} \tag{4.2}$$

4.1.2 Aspect-based Sentiment Analysis with Relation Extraction

We proposed a new method for identifying the sentiment category of a given aspect based on the aspect-opinion relations. In this thesis, the aspect-opinion relation is defined as follows: there exists the aspect-opinion relation between a word (or phrase in general) A and O in a sentence if O expresses an opinion of A (an aspect of a certain entity). The method supposes that the opinion words related to the aspect will more influence the polarity of it. Identification of the aspect-opinion relations in the sentence can help to improve the prediction of sentiment categories of the given aspect. In other words, aspect-opinion relation extraction enables us to distinguish opinion words of the target aspect and other aspects.

For a given sentence, the aspect-opinion relations were extracted by using the tree kernel based on constituent and dependency trees. The detail algorithms of aspect-opinion relation extraction will be described later. Then, we put more weight on the important opinion words in the sentiment score of the aspect as shown in Equation (4.3). Specifically, if there is a relation between an aspect and opinion, the weight $weight(a, ow)$ is set to 2, otherwise 1.

$$sentimentValue(a_i) = \sum_{j=1}^{|OW|} weight(a_i, ow_j) \cdot \frac{opinionValue(ow_j)}{distance(a_i, ow_j)} \tag{4.3}$$

$$weight(a, ow) = \begin{cases} 2 & \text{if } r(a, ow) = 1 \\ 1 & \text{otherwise} \end{cases} \tag{4.4}$$

Figure 4.1 shows the architecture of our model. Intuitively, given the input sentence "Food in Egypt is delicious" where "food" is the aspect, the word "delicious" may greatly contribute to determine the polarity of the word "food" because there is a relation between

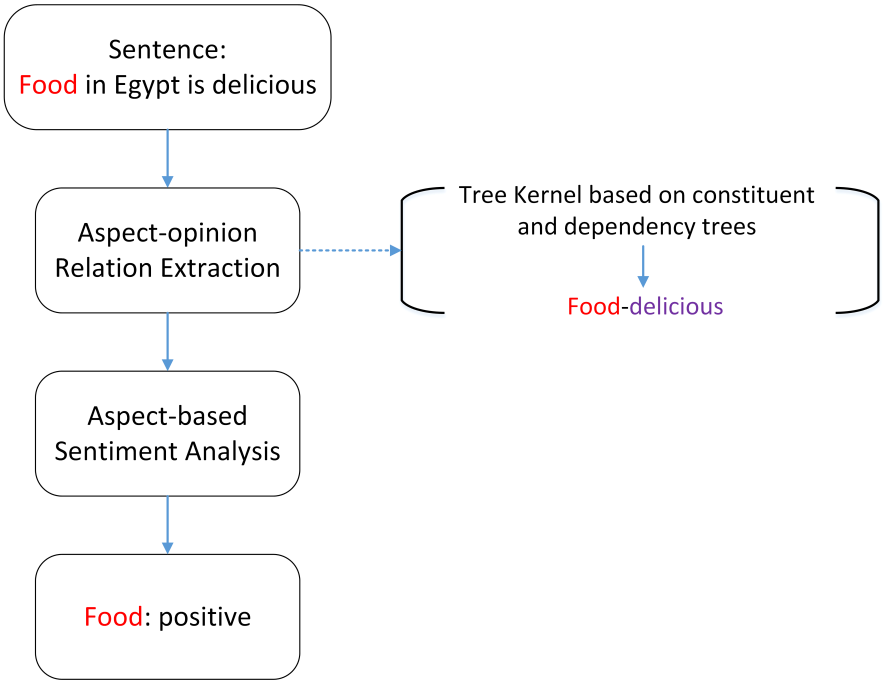

Figure 4.1: Architecture of Aspect-based Sentiment Analysis Using Tree Kernel based Aspect-Opinion Relation Extraction Module

these two words. The aspect-opinion relation extraction will extract the relation "food-delicious" by using tree kernel based on the constituent and dependency trees. Then, this relation will be integrated into the aspect-based sentiment analysis model to predict the sentiment category of the aspect "food" as positive.

The rest of this subsection presents the details of the proposed method for the aspect-opinion relation extraction, following brief introduction of relation extraction task.

Relation Extraction

Relation extraction is a task of finding relations between pairs of entities in texts. Many approaches have been proposed to learn the relations from texts. Among these approaches, kernel methods have been increasingly used for the relation extraction [92, 93, 94, 95, 96]. The main benefit of kernel methods is that they can exploit a huge amount of features without an explicit feature representation [97, 21, 22]. In the relation extraction task, many kinds of relations, from general to specific ones, are considered. Here we focuses on aspect-opinion relation, which is a relation between an aspect of an entity (eg. a price of a PC) and an opinion word or phrase that expresses evaluation on that aspect. It is still an open question if the kernel methods also work well for aspect-opinion relation extraction.

Some previous work used the dependency tree kernels for general relation extraction [92, 93, 95]. In these researches, they tried to extract all of the predefined relations in a given sentence. The predefined relations are *person-affiliation*, *organization-location* and so on. Nguyen et al. used tree kernel based on the constituent, dependency and sequential structures for relation extraction [98]. They focused on seven relation types such as

Table 4.1: Features used in SVM-B

Feature	**Values**
Position of opinion word in sentence	{start, end, other}
Position of aspect word in sentence	{start, end, other}
The distance between opinion and aspect	{1, 2, 3, 4, other}
Whether opinion and aspect have direct dependency relation	{True, False}
Whether opinion precedes aspect	{True, False}
Part of Speech (POS) of opinion	Penn Treebank Tagset
POS of aspect	Penn Treebank Tagset

person-affiliation in ACE corpus, which was well-known as a dataset for general relation extraction. However, aspect-opinion relation was not considered in these researches. For the aspect-based sentiment analysis, it is very important to know whether there is a relation between an aspect and opinion word. To the best of our knowledge, there is a lack of researches trying to use tree kernel for aspect-opinion relation extraction.

Wu et al. proposed a phrase dependency parsing for extracting relations between product features and expression of opinions [94]. Their tree kernel is based on a phrase dependency tree converted from an ordinary dependency tree. However, they did not apply this model for calculating a sentiment score for a given aspect.

Bunescu and Mooney extracted the shortest path between two entities in a dependency tree to identify the relation between them [99]. The dependency kernel was calculated based on this shortest path. They suggested that the shortest path encodes sufficient information for relation extraction.

Kobayashi et al. combined contextual and statistical clues for extracting aspect-evaluation and aspect-of relations [100]. Since the contextual information is domain-specific, their model cannot be easily used in other domains.

Aspect-Opinion Relation Extraction

For a given sentence where an aspect phrase and opinion phrase have been already identified, we will determine whether there is a relationship between the aspect and opinion phrase. To achieve this goal, four supervised machine learning methods will be presented. One is Support Vector Machine (SVM) with a linear kernel and the others are SVM with tree kernels.

SVM-B: a baseline model

SVM has long been recognized as a method that can efficiently handle high dimensional data and has been shown to perform well on many applications such as text classification [97, 21, 101, 102]. A set of features used for training SVM is shown in Table 4.1. They are common features used for relation extraction. Because this model was also used in previous work [100, 94], we chose it as a baseline model to compare with other methods.

CTK: Constituent Tree based Tree Kernel:

Tree kernel for the constituent tree has been used successfully in many applications. Various tree kernels have been proposed such as subtree kernel [103] and subset tree kernel

[104]. We applied the subtree kernel for this research. Figure 4.2 shows an example of a constituent tree for the sentence "It has excellent picture quality and color."

Given a constituent tree of a sentence, we represented each $r(e_1, e_2)$, aspect-opinion relation between the aspect entity e_1 and opinion entity e_2, as a subtree T rooted as the lowest common parent of e_1 and e_2. Notice that the aspect and opinion entity can be phrases in general. The subtree T must contain all of the words in these phrases. For example, the relation between the aspect "picture quality" and opinion "excellent" in Figure 4.2 is represented by the subtree rooted at "NP" node [1], which is the lowest common parent of "picture", "quality" and "excellent" node. The main idea of this tree kernel is to compute the number of the common substructures between two tree T_1 and T_2 which represent two relation instances. The kernel between two trees T_1 and T_2 is defined as in Equation (4.5).

$$K(T_1, T_2) = \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} C(n_1, n_2) \tag{4.5}$$

N_1 and N_2 are the set of the nodes in T_1 and T_2. $C(n_1, n_2)$ is the number of common subtrees of two trees rooted at node n_1 and n_2. It is calculated as follows:

1. If n_1 and n_2 are pre-terminals with the same POS tag: $C(n_1, n_2) = \lambda$
2. If the production rules at n_1 and n_2 are different: $C(n_1, n_2) = 0$
3. If the production rules at n_1 and n_2 are the same:

$$C(n_1, n_2) = \lambda \prod_{j=1}^{nc(n_1)} (1 + C(ch(n_1, j), ch(n_2, j)))$$

where $nc(n_1)$ is the number of the children of n_1 in the tree. $ch(n_i, j)$ is the j^{th} child-node of n_i. Since the production rules at n_1 and n_2 are the same, $nc(n_1) = nc(n_2)$. We set $\lambda = 0.5$ in our experiment.

Finally, since the value of $K(T_1, T_2)$ will depend greatly on the size of the trees T_1 and T_2, we normalize the kernel as in Equation (4.6).

$$K'(T_1, T_2) = \frac{K(T_1, T_2)}{\sqrt{K(T_1, T_1) K(T_2, T_2)}} \tag{4.6}$$

DTK: Dependency Tree based Tree Kernel:
A dependency tree kernel has been proposed by Culotta and Sorensen for general relation extraction [92]. This thesis applies it for aspect-opinion relation extraction. Given a dependency tree of a sentence, we represent each relation $r(e_1, e_2)$ as a subtree T rooted as the lowest common parent of the aspect e_1 and opinion e_2. For example, the relation between the aspect "picture quality" and opinion "excellent" in Figure 4.3 is the subtree rooted at "quality" node, which is the lowest common parent of "picture", "quality" and "excellent" node.

[1] It is denoted by the circle in Figure 4.2.

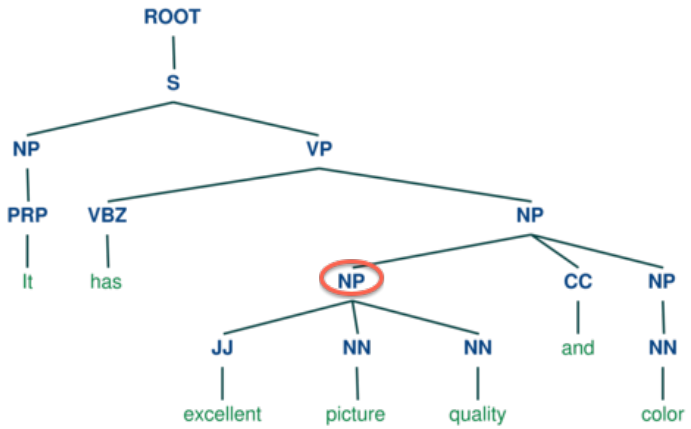

Figure 4.2: An Example of Constituent Parse Tree

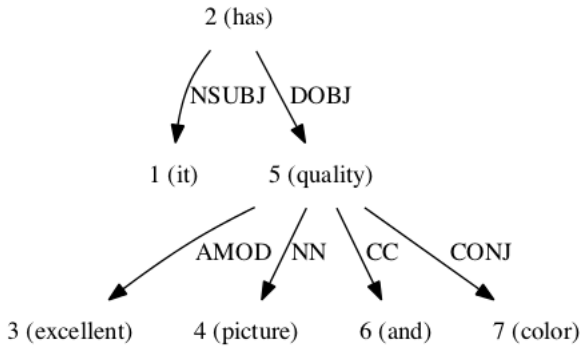

Figure 4.3: An Example of Dependency Tree

A subtree T of a relation instance can be represented as a set of nodes $\{n_0, \cdots, n_t\}$. Each node n_i is augmented with a set of features $f(n_i) = \{v_1, \cdots, v_d\}$. They are subdivided into two subsets $f_m(n_i)$ (features used for matching function) and $f_s(n_i)$ (for similarity function). A matching function $m(n_i, n_j) \in \{0, 1\}$ in Equation (4.7) checks if $f_m(n_i)$ and $f_m(n_j)$ are the same. A similarity function $s(n_i, n_j)$ in $(0, \infty]$ in Equation (4.8) evaluates the similarity between $f_s(n_i)$ and $f_s(n_j)$.

$$m(n_i, n_j) = \begin{cases} 1 & \text{if } f_m(n_i) = f_m(n_j) \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

$$s(n_i, n_j) = \sum_{v_q \in f_s(n_i)} \sum_{v_r \in f_s(n_j)} C(v_q, v_r) \tag{4.8}$$

In Equation (4.8), $C(v_q, v_r)$ is a compatibility function between two feature values as:

$$C(v_q, v_r) = \begin{cases} 1 & \text{if } v_q = v_r \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

For two given subtrees T_1 and T_2 which represent two relation instances with root nodes r_1 and r_2, the tree kernel $K(T_1, T_2)$ is defined as in Equation (4.10):

$$K(T_1, T_2) = \begin{cases} 0 & \text{if } m(r_1, r_2) = 0 \\ s(r_1, r_2) + K_c(r_1[c], r_2[c]) & \text{otherwise} \end{cases} \tag{4.10}$$

where K_c is a kernel function over children. Let $\mathbf{a}$ and $\mathbf{b}$ be sequences of children nodes' indices of node n_i and n_j, respectively. We denote the length of $\mathbf{a}$ by $l(\mathbf{a})$. K_c is defined as Equation (4.11). $n_i[\mathbf{a}]$ stands for the subtree consisting of children indicated by $\mathbf{a}$, while $n_i[a_h]$ is h^{th} child of n_i. In this equation, we consider the contiguous kernel enumerating children subsequences that are not interrupted by not matching nodes. In our experiment, λ is set to 0.5.

$$K_c(n_i[c], n_j[c]) = \sum_{\mathbf{a},\mathbf{b}, l(\mathbf{a})=l(\mathbf{b})} \lambda^{l(\mathbf{a})} K(n_i[\mathbf{a}], n_j[\mathbf{b}]) \prod_{h=1}^{l(\mathbf{a})} m(n_i[a_h], n_j[b_h]) \tag{4.11}$$

Finally, we also normalize the kernel as in Equation (4.6).

The augmented features are shown in Table 4.2. Note that *Label*, *isAspectNode* and *isOpinionNode* are used for matching between two nodes, while the rest is used for measuring the similarity of them.

CTK + DTK: Combination of Two Kernels:

We proposed a new tree kernel based on the combination of two kernels CTK and DTK for aspect-opinion relation extraction. That is, we try to utilize the information from both the constituent and dependency tree. Equation (4.12) defines the combined kernel

Table 4.2: Features for Each Node in the Dependency Tree

	Feature	**Values**
f_m	$Label$	Penn Treebank POS Tagset
	$isAspectNode$	{0,1}
	$isOpinionNode$	{0,1}
f_s	NER	StanfordCoreNLP Name Entity Tagset
	$relationToParentNode$	StanfordCoreNLP Dependency Relation
	$LabelofParentNode$	Penn Treebank POS Tagset
	$NERofParentNode$	StanfordCoreNLP Name Entity Tagset

function.

$$K_{CTK+DTK}(T_1, T_2) = K_{CTK}(T_1, T_2) + K_{DTK}(T_1, T_2) \tag{4.12}$$

$K_{CTK}(T_1, T_2)$ and $K_{DTK}(T_1, T_2)$ are the CTK and DTK tree kernels, respectively. Since the summation of two kernels is valid, $K_{CTK+DTK}$ is obviously a valid kernel.

4.1.3 RNN: Recursive Neural Network

As discussed in Subsection 2.1.2, RNN is a deep network to compute the parent representations from the child nodes. It represents each word or phrase in the binary tree as a d-dimensional vector. From bottom to up, the representations of parent nodes are calculated by linear combination of child vector representations. In this research, we employ RNN as a baseline model to compare with our models. Dong et al shows an algorithm to convert a dependency tree to a binary tree to apply RNN for aspect-based sentiment analysis [60]. We also employed the same method. The details of the conversion algorithm will be explained in the next subsection 4.1.4.

4.1.4 AdaRNN: Adaptive Recursive Neural Network

Adaptive Recursive Neural Network (AdaRNN) is a recursive neural network with multi-compositionality layers [60, 61]. It employed more than one composition functions and adaptively chooses them depending on the context and linguistic tags.

Figure 4.4 shows the example of AdaRNN. For a sentence "windows is better than ios", where "windows" and "ios" are the targets we want to predict the sentiment categories. AdaRNN converts the dependency tree into the binary tree for each target word. The root word of the binary tree is used as the vector representation for the target word. Then, this representation is fed to a sofmax classifier such as logistic regression to predict the sentiment category for the aspect.

The algorithm shown in Figure 4.5 explains the function to convert from a tree to a binary tree with a target word. The input of this function is a list of edges E in a tree and a target word v_t. The output of this function is the node v which is the root node of the binary tree. Precisely, the returned node v includes pointers to the left and right child in the form of $[l, r]$, where l and r are also root nodes of binary subtrees or leaf nodes.

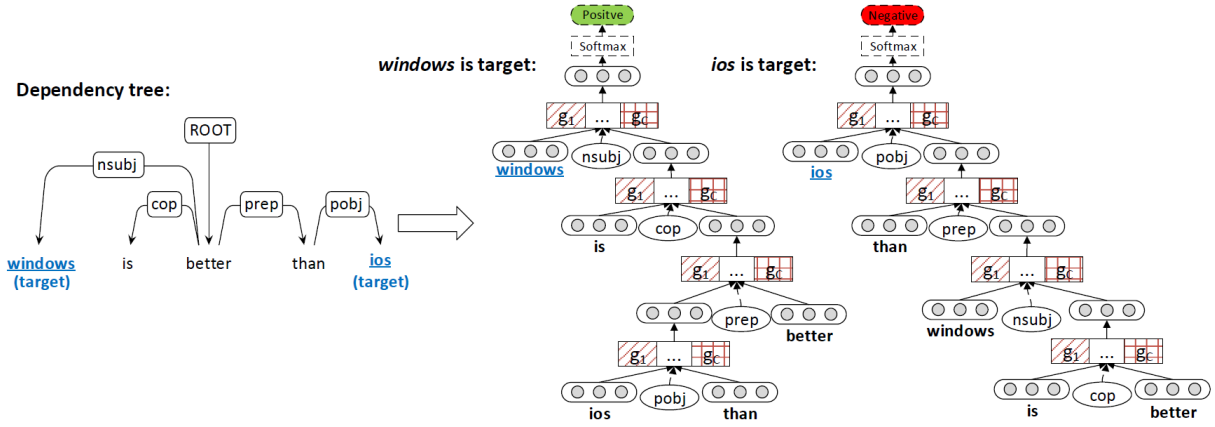

Figure 4.4: Example of AdaRNN Model

```
Function convert(E, v_t):
    v ← v_t
    for v_i → v_t, v_t → v_i in E do
        if v_t → v_i then
            E' ← E \ {v_t → v_i}
            w ← [convert(E', v_i), v]
        else
            E' ← E \ {v_i → v_t}
            w ← [v, convert(E', v_i)]
        end
        v ← w
    end
    return v
end
```

Figure 4.5: Algorithm for Converting a Tree to a Target Binary Tree

In other words, the node v is represented as the nested brackets. Let $a = [b, c]$ denote a parent node a with left child b and right child c. By looping over edges containing the target vertex v_t, the function creates an intermediate node, depending on whether v_t is the head in this relation edge or not. This function will be called recursively to create the binary tree. For example, supposing that E is a set of edges in the dependency tree in Figure 4.4, $convert(E, windows)$ returns $[windows, [is, [[ios, than], better]]]$, while $convert(E, ios)$ returns $[ios, [than, [windows, [is, better]]]]$. Graphical representations of these binary trees are shown in Figure 4.4.

AdaRNN used n composition functions $g_1, \cdots, g_n$. These functions are defined as the linear combination functions of the vector representations of left child v_l and right child v_r as in Equation (4.13).

$$W \begin{bmatrix} v_l \\ v_r \end{bmatrix} + b \tag{4.13}$$

where $W \in \Re^{d \times 2d}$ is the composition matrix and $b \in \Re^d$ is the bias vector.

The vector representation of parent node v is computed from the vector representation of left child v_l and right child v_r as in Equation (4.14).

$$v = f\left(\sum_{i=1}^{n} P(g_i|v_l, v_r, e_{in}) g_i(v_l, v_r)\right) \tag{4.14}$$

where f is a nonlinear function such as *tanh* or *sigmoid*. e is the external feature vector. $P(g_i|v_l, v_r, e)$ is the probability of function g_i given the child vectors v_l, v_r and external feature vector e.

The probabilities of the composition functions are defined as in Equation (4.15).

$$\begin{bmatrix} P(g_1|v_l, v_r, e) \\ \cdots \\ P(g_n|v_l, v_r, e) \end{bmatrix} = softmax\left(\beta S \begin{bmatrix} v_l \\ v_r \\ e \end{bmatrix}\right) \tag{4.15}$$

where $\beta \in \Re$ is a hyper-parameter which are the trade-offs between uniform selection and maximum selection, and $S \in \Re^{n \times (2d+|e|)}$ is the matrix.

The value of $\beta = 1$ seems to be better than other values [60]. Therefore, in this research, we also employ $\beta = 1$ for AdaRNN and PhraseRNN described in 4.1.5.

4.1.5 PhraseRNN: Phrase Recursive Neural Network

We proposed a new supervised method, PhraseRNN, for aspect-based sentiment analysis. PhraseRNN is constructed based on the recursive neural network. In this model, a representation of an aspect will be learnt from a target dependent tree structure constructed by combining the constituent and dependency tree. The intermediate parent node combined the two child nodes. This will be recursively constructed until reaching the root node of tree structure which is the representation of the aspect.

Formally, given a sentence with an aspect a, PhraseRNN will construct a target dependent binary tree structure corresponding to this aspect. From the leaf nodes of a tree, PhraseRNN will build the representation for aspect a following the bottom to up fashion. To construct the tree structure, PhraseRNN uses two algorithms as follows:

1. Convert the dependency tree and the phrases in the constituent tree to the phrase dependency tree by the algorithm shown in Figure 4.8.

2. Convert the phrase dependency tree to the target dependent binary phrase dependency tree at both inner-phrase and outer-phrase level by the algorithm shown in Figure 4.10.

Figure 4.6 and 4.7 show the dependency and constituent tree for the example sentence "Except the design, the phone is bad for me." In this sentence, "design" and "phone" are aspects, and we want to determine the polarities of them.

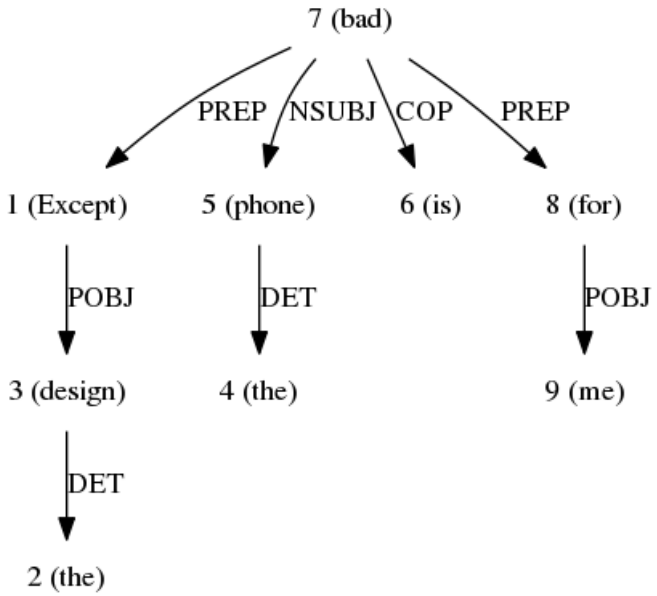

Figure 4.6: A Dependency Tree for an Example Sentence

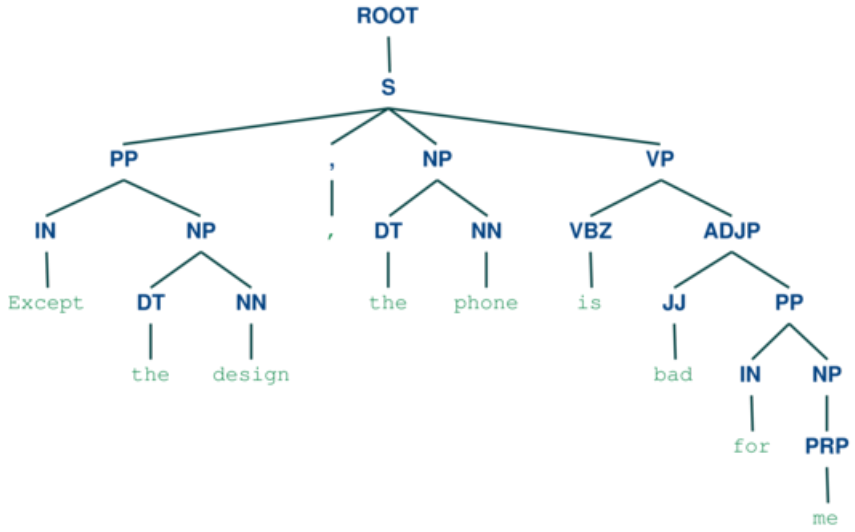

Figure 4.7: A Constituent Tree for an Example Sentence

Converting to a Phrase Dependency Tree

First of all, we identify the non-overlapping constituents (noun phrases, verb phrases, preposition phrases and so on) in the sentence. This could be considered as a shallow chunking task. For example, from the example sentence "Except the design, the phone is bad for me.", three phrases are identified as the chunks: PP[Except the design], NP[the phone] and VP[is bad for me]. In this research, we use a simple algorithm to extract these phrases from the constituent tree. The algorithm begins from the root node of the constituent tree, and travels from top to bottom. If a node has at least two meaningful child nodes, we use these child nodes as the phrases. Punctuation nodes such as '.', ',', ':' and so on will be considered as non-meaningful nodes. Square brackets indicate all words governed by the phrase.

For instance, from the constituent tree of the sentence in Figure 4.7, the algorithm starts from ROOT node and continues through S node which has three meaningful child nodes: PP, NP and VP. The words 'Except', 'the' and 'design' are leaf nodes in the subtree of PP phrase, these words will be extracted as PP[Except the design]. This procedure is applied to NP and VP nodes to extract the words in these phrases. Let K be the number of phrases. The list of extracted phrases is represented as $P = \{p_1, \cdots, p_K\}$ where p_i is a phrase consisting of the non-terminal symbol and words in this phrase. The algorithm outputs the list of phrases P = {PP[Except the design], NP[the phone], VP[is bad for me]}.

Given a dependency tree and a list of phrases, the algorithm in Figure 4.8 will create a phrase dependency tree. The input of this algorithm is a dependency tree $T = (V, E)$ consisting of a set of vertices $V = \{v_1, \cdots, v_{|V|}\}$ and a set of relation edges $E = \{(r_{ji}, v_i, v_j)\}$ between two vertices, and a list of phrases $P = \{p_1, \cdots, p_K\}$ extracted from the constituent tree. The output will be a phrase dependency tree $pT = (pV, pE)$ where $pV = \{T_1, \cdots, T_K\}$, each $T_i = (V_i, E_i)$ is a subtree, and $pE = \{(r_{ji}, T_i, T_j)\}$ is a list of relations between two subtrees.

The algorithm will firstly create subtrees $T_1, \cdots, T_K$ as vertices in the phrase dependency trees pT. The vertices in subtree T_i are vertices in phrase p_i. Then, by looping over edge e_i in the dependency tree T, if e_i connects two vertices in a phrase p_k, we add this edge to the list of edges of subtree T_k. Otherwise, if e_i connects two vertices in two phrases p_k and p_l, we add this edge to the list of edges pE as the relation of two subtrees T_k and T_l.

For example, with the dependency tree in Figure 4.6 and the extracted phrases P = {PP[Except the design], NP[the phone], VP[is bad for me]}, the algorithm will output a phrase dependent tree in Figure 4.9. The vertices of this phrase dependent tree are three subtrees PP, VP and NP corresponding to the list of phrases. There are two relations between subtrees. The relation $PREP$ is between PP and VP subtrees where VP subtree is the head. The other is $NSUBJ$ between NP and VP subtrees where VP subtree is the head. Each of PP, VP and NP subtrees has vertices as words in the corresponding phrases. For example, the subtree PP has three vertices 'Except', 'design' and 'the'. The

Input: dependency tree $T = (V, E)$, phrase list $P = \{p_1, \cdots, p_K\}$
Output: phrase dependency tree: $pT = (pV, pE)$ where $pV = \{T_1, \cdots, T_K\}$, $T_i = (V_i, E_i)$ and $pE = \{(r_{ji}, T_i, T_j)\}$

for *each phrase* $p_i \in P$ **do**
  $V_i \leftarrow \{v_j | v_j \in p_i\}$
end
for *each edge* $e_i = (r_{nm}, v_m, v_n) \in E$ **do**
  $v_m \in p_k$, $v_n \in p_l$
  if $k = l$ **then**
    $E_k \leftarrow E_k + (r_{nm}, v_m, v_n)$
  else
    $pE \leftarrow pE + (r_{nm}, T_k, T_l)$
  end
end

Figure 4.8: Algorithm for Converting from Dependency Tree to Phrase Dependency Tree

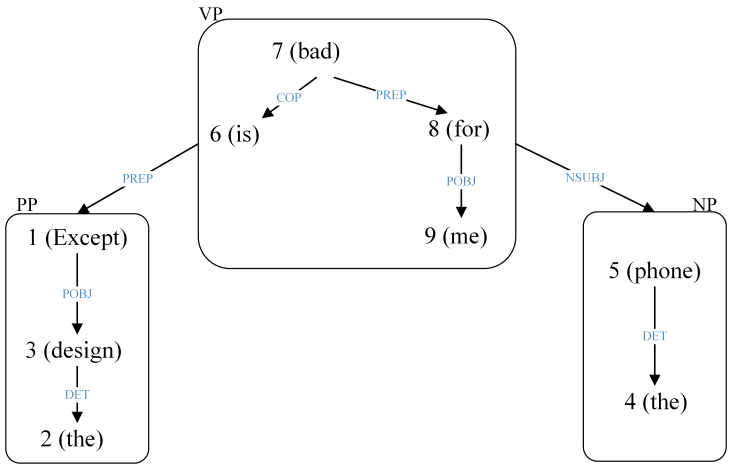

Figure 4.9: An Example of Phrase Dependency Tree

advantage of the phrase dependent tree is that it groups the relations in the same phrase and has the relations between groups. In other words, there are inner-phrase relations and outer-phrase relations in the phrase dependent tree.

Converting a Phrase Dependency Tree to a Target Dependent Binary Phrase Dependency Tree

After the phrase dependency tree is constructed, it is transformed into a target dependent binary tree bpT as shown in the algorithm in Figure 4.10. The input of the algorithm is a phrase dependency tree $pT = (pV, pE)$ and a target word v_t (the aspect word we want to predict the sentiment category). The output is the root node of a binary tree bpT. Basically, the leaves of the binary tree bpT are binary subtrees $bT_1, \cdots, bT_K$ which are the binary versions of subtrees $T_1, \cdots, T_K$. The leaves of binary subtree bT_i are the words in

Input: phrase dependency tree: $pT = (pV, pE)$, target v_t
where $pV = \{T_1, \cdots, T_K\}$, $T_i = (V_i, E_i)$ and $pE = \{(r_{ji}, T_i, T_j)\}$
Output: target dependent binary phrase dependency tree: bpT

```
1  for T_i = (V_i, E_i) ∈ pV do
2      if v_t ∈ V_i then
3          h ← v_t
4      else
5          h ← vertex having no head in E_i
6      end
7      bT_i ← convert(E_i, h)
8  end
9  T_{v_t} ← T_i that contains v_t
10 bpT ← convert(pE, T_{v_t})
11 Replace all T_i in bpT with bT_i
```

Figure 4.10: Algorithm for Converting from Phrase Dependency Tree to Target Dependent Binary Phrase Dependency Tree

phrase p_i. Each node in the binary trees or subtrees has two pointers. One points to the left child node, the other to the right child node.

The algorithm firstly converts each of subtree T_i in the phrase dependency tree pT into the binary subtree bT_i by *convert* function shown in Figure 4.5. After the conversion of T_i, the overall phrase dependency tree is converted to a binary tree at line 10 in Figure 4.10. In this step, subtrees T_i are treated as the nodes. Finally, every T_i in bpT are replaced with the converted binary tree bT_i.

By converting from the phrase dependency tree in Figure 4.9, the binary phrase dependency trees for the target aspects "design" and "phone" are shown in Figure 4.11 and 4.12, respectively. Notice that, each aspect in the sentence has a different binary phrase dependency tree.

Constructing the Parent Representation from Child Nodes

Each node in the binary tree is represented as a d-dimensional vector of real value. In this research, we used the pre-trained Google News dataset by word2vec [2] algorithms. Each word is represented as a 300-dimensional vector in this pre-trained dataset.

PhraseRNN uses two kinds of composition function $G = \{g_1, \cdots, g_n\}$ for inner-phrase and $H = \{h_1, \cdots, h_m\}$ for outer-phrase. n and m are the number of functions in G and H, respectively. We denote v_{in} as a vector representation of a node in the binary subtree bT_i. Let v_{out} be a vector representation of a node in the binary tree bpT. v_l and v_r are vector representations of left and right children of node v_{in} or v_{out}. $g_1, \cdots, g_n$ and $h_1, \cdots, h_m$ are defined as the linear combination functions of left and right children as in Equation (4.13).

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

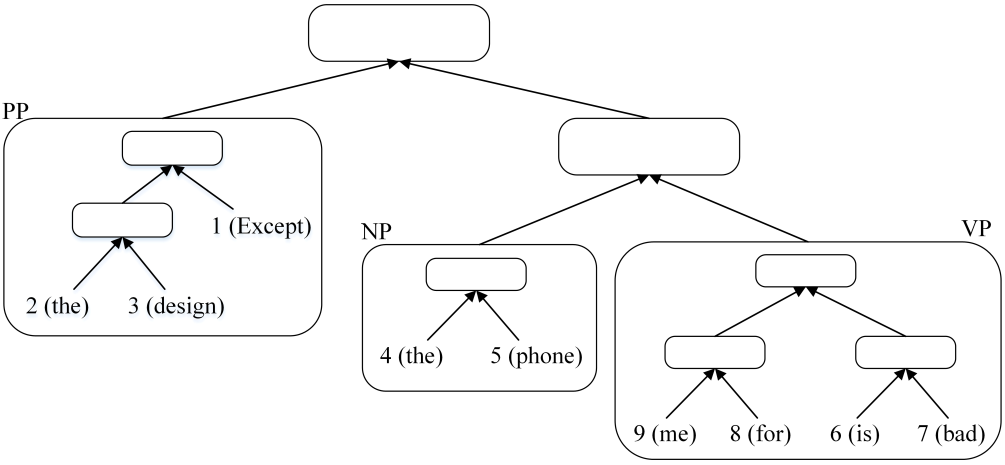

Figure 4.11: Target Dependent Binary Phrase Dependency Tree for the Target Aspect "design"

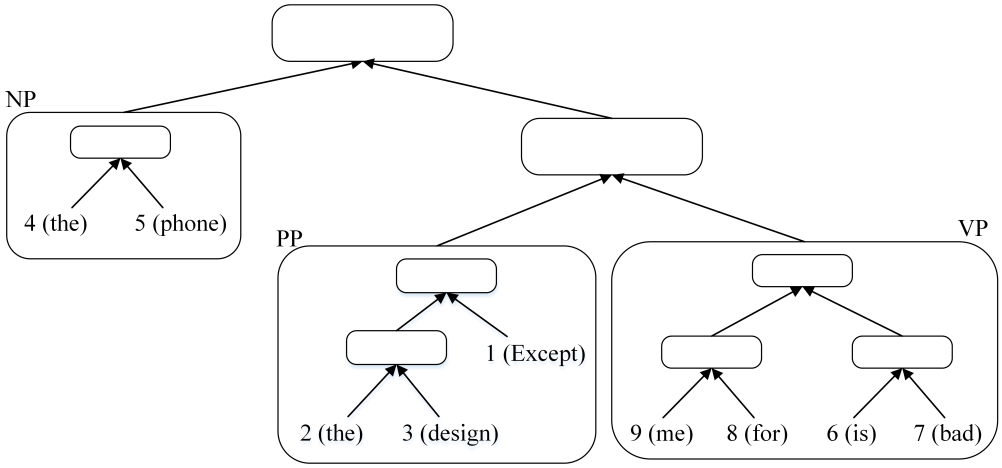

Figure 4.12: Target Dependent Binary Phrase Dependency Tree for the Target Aspect "phone"

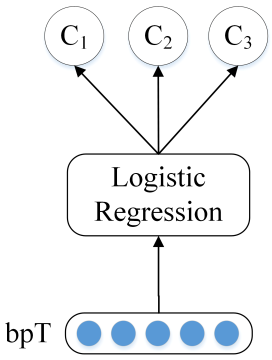

Figure 4.13: Prediction Using Logistic Regression

We calculated the parent node v_{in} of left child v_l and right child v_r in the binary subtree bT_i as in Equation (4.16):

$$v_{in} = f\left(\sum_{i=1}^{n} P(g_i|v_l, v_r, e_{in}) g_i(v_l, v_r)\right) \tag{4.16}$$

where f is a nonlinear function such as *tanh* or *sigmoid*. e_{in} is the external feature vector. $P(g_i|v_l, v_r, e_{in})$ is the probability of function g_i given the child vectors v_l, v_r and external feature vector e_{in}. We define the probabilities of these composition functions as in Equation (4.17).

$$\begin{bmatrix} P(g_1|v_l, v_r, e_{in}) \\ \cdots \\ P(g_n|v_l, v_r, e_{in}) \end{bmatrix} = softmax\left(\beta R \begin{bmatrix} v_l \\ v_r \\ e_{in} \end{bmatrix}\right) \tag{4.17}$$

where $\beta \in \Re$ is a hyper-parameter, and $R \in \Re^{n \times (2d+|e_{in}|)}$ is the matrix.

In the target dependent binary phrase dependency tree bpT, the parent node v_{out} combined of two children v_l and v_r is computed as in Equation (4.18).

$$v_{out} = f\left(\sum_{i=1}^{m} P(h_i|v_l, v_r, e_{out}) h_i(v_l, v_r)\right) \tag{4.18}$$

where $P(h_i|v_l, v_r, e_{out})$ is the probability of function h_i given the child vectors v_l, v_r and external feature vector e_{out}. We define the probabilities of these composition functions as:

$$\begin{bmatrix} P(h_1|v_l, v_r, e_{out}) \\ \cdots \\ P(h_m|v_l, v_r, e_{out}) \end{bmatrix} = softmax\left(\beta S \begin{bmatrix} v_l \\ v_r \\ e_{out} \end{bmatrix}\right) \tag{4.19}$$

where $\beta \in \Re$ is a hyper-parameter, and $S \in \Re^{m \times (2d+|e_{out}|)}$ is the matrix.

The external features $e_i = \{Label_l, Label_r, DepType_i\}$ of the node v_i consists of three types of features. $Label_l$ and $Label_r$ are the label of left and right nodes, respectively. If node v_l is a leaf word, it is the POS of word v_l. If v_l is the intermediate node, this is the non-terminal symbol of the lowest common parent of descendants of v_l in the constituent tree. For example, the *Label* of the node combined from "the" and "design" in Figure 4.11 is NP which is the lowest common parent of these two words in the constituent tree in Figure 4.7. $DepType_i$ is the dependency relation for node v_i. If left and right nodes are leaf nodes, this is the direct relation in the dependency tree between these two nodes. If at least one of two child nodes is the intermediate node, $DepType_i$ is the relation between head words of the left and right nodes. For example, in Figure 4.11, let a be the parent of "is" and "bad", b is the parent of "for" and "me", c is the parent of a and b. *DepType* of a is COP which is the dependency relation between two child nodes with head of relation is "bad". Similarly, *DepType* of b is POBJ. *DepType* of c is $PREP$ which is the

dependency relation between "bad" and "for". Then, feature e_i of node v_i is converted to a binary feature vector. For example, if $POS_l = k$, then $e_i[k] = 1$, others to 0.

The vector representation of the root node in the output binary tree will be used as the vector representation for the aspect word. Then, this vector representation will be inputted to a logistic regression in order to predict the sentiment category for the target aspect as in Figure 4.13. We suppose a batch training data consist of B training instances $\{(x^{(1)}, t^{(1)}), \cdots, (x^{(B)}, t^{(B)})\}$, where $x^{(b)}$ and $t^{(b)}$ are the aspect and its sentiment category of b instance. Let $y^{(b)}$ be the predicted sentiment category for aspect $x^{(b)}$ by PhraseRNN. The goal is to minimize the loss function which is the mean of negative log likelihood plus with L2 regularization penalty in a batch training set as in Equation (4.20).

$$L = -\frac{1}{B}\sum_{b=1}^{B} log(P(y^{(b)} = t^{(b)}|x^{(b)}, \theta)) + \lambda \sum_{\theta_i \in \theta} \parallel \theta_i \parallel^2 \tag{4.20}$$

where λ is a constant controlling the degree of penalty, θ is all the parameters in the model.

Stochastic gradient descent is used to optimize the loss function. Backpropagation was employed to propagate the errors from the top node to the leaf nodes. The derivatives of parameters are used to update the parameters.

We consider the following variations of PhraseRNN:

1. PhraseRNN-1: use only a global weight matrix: $G = H = g_1$
2. PhraseRNN-2: use two global weight matrices. One for inner-phrase, the other for outer-phrase: $G = g_1$ and $H = h_1$
3. PhraseRNN-3: use a list of global weight matrices: $G = H = \{g_1, \cdots, g_n\}$
4. PhraseRNN-4: use two list of global weight matrices. One for inner-phrase, the other for outer-phrase: $G = \{g_1, \cdots, g_n\}$ and $H = \{h_1, \cdots, h_m\}$

4.2 Methods for Aspect Category Polarity Identification

Given a sentence, this section examines the models determining the polarity (positive, neutral and negative) of each aspect category. Let us review what the task of aspect category polarity identification is. Aspect category is a set of pre-defined aspects for a certain type of a product or service. They are typically coarser than the aspect terms. In addition, they do not necessarily occur as terms in the given sentence. For example, "price" and "food" are typical aspect categories for the restaurants. In the sentence "The restaurant was expensive, but the menu was great," the models should identify the aspect category "price" as negative and "food" as positive.

The following two topic models are applied for the aspect category polarity identification.

1. JST:
   The JST model [66] was considered as the state of the art of topic modeling for sentiment analysis. Therefore, we use JST as the baseline model. To get the sentiment of each aspect category in a document, we get the most highly associated aspect category and its sentiment for each word in that document. Then the most frequent sentiment of each aspect category in the document is chosen for each aspect category.

2. TSLDA:
   We use our TSLDA model for the sentiment analysis and compare it to JST. We firstly get the most highly associated aspect category and sentiment category for each sentence in the document. Then the most frequent sentiment in all sentences in the document is chosen for each aspect category.

The disadvantage of topic models is that the hidden topics could not be explained verbally. They only provide probabilistic distributions over words. We could consider these hidden topics as the aspect categories. However, it is uncertain which aspect category the hidden topic corresponds to. Furthermore, the sentiments in JST and TSLDA are also hidden variables. Therefore, we need to map the inferred topics/sentiment to human topics/sentiments. Here the human topic refers to the pre-defined aspect category and human sentiment stands for either positive, negative or neutral. In the next two subsections, we will explain our proposed methods for inferring the human topics/sentiments to the hidden topics/sentiments.

4.2.1 Mapping Inferred Topics to Human Topics

We propose a method to automatically guess the correspondence between inferred and human topics. Because inferred topics are distributions over words, we will convert human topics as distributions over words. Then, we measure the divergence between these two distributions by using Kullback-Leibler divergence. An inferred topic will be mapped to a human topic with the smallest divergence.

Converting Human Topics to Distributions Over Words

Let $V = \{w_1, \cdots, w_N\}$ as the vocabulary. Dice Coefficient $dice(w_i, w_j)$ between two words w_i and w_j is defined as Equation (4.21).

$$dice(w_i, w_j) = \frac{2|w_i \cap w_j|}{|w_i| + |w_j|} \tag{4.21}$$

where $|w_i \cap w_j|$ is the number of times that two words w_i and w_j appearing in the same sentence in a context window. The window size is set to 10. $|w_i|$ and $|w_j|$ are the number

Input: Training dataset, a list S of seed words of a human topic, vocabulary V
Output: A distribution over words $H = \{p_1, \cdots, p_N\}$

for *each word* $w_i \in V$ **do**
  for *each word* $w_j \in S$ **do**
    Calculate $dice(w_i, w_j)$;
  end
  Set $p_i = \max_{w_j \in S} dice(w_i, w_j)$;
end
Normalize: $p_i = \frac{p_i}{\sum_{k=1}^{N} p_k}$

Figure 4.14: Algorithm for Converting a Human Topic to a Distribution over Words

of times that words w_i and w_j appearing in the dataset. The range of dice coefficient is $0 \leq dice(w_i, w_j) \leq 1$. Intuitively, the higher co-occurrence of two words, the higher value of dice coefficient. If two words do not co-occur at all, the dice coefficient will be 0. On the other hand, if two words always occur together, the dice coefficient will be 1.

The algorithm in Figure 4.14 shows our way to convert a human topic to a distribution over words. First, we manually define a list of seed words that are representative words for the human topic. For example, the human topic "display" in "Laptops" domain can have a list of seed words $\{display, monitor, screen\}$. The algorithm outputs a distribution over words $H = \{p_1, \cdots, p_N\}$ where p_i is the probability of word w_i in that topic. The probability p_i is calculated as the max value of dice coefficient between word w_i and seed words. Note that the seed words are also included in the vocabulary list. Because the seed words are highly associated with that topic, the probability of the seed word is designed to be the highest value. Finally, we normalize these probability values to make sure that $\sum_{k=1}^{N} p_k = 1$.

Divergence between Two Distributions

Next, we need to evaluate how an inferred topic I is similar to a human topic H. Kullback-Leibler divergence is a non-symmetric measure of the difference between two probability distributions I and H. Specifically, Kullback-Leibler divergence of H from I, denoted $KL(I||H)$, is a measure of the information loss when H is used to approximate I. For a discrete probability distribution, Kullback-Leibler divergence of H from I is defined as Equation (4.22):

$$KL(I||H) = \sum_i I(i) ln \frac{I(i)}{H(i)} \tag{4.22}$$

where $I(i), H(i)$ are the probability of i^{th} element in I and H, respectively.

The $KL(I||H)$ is always non-negative and equals 0 if and only if $I = H$. The closer I and H are, the smaller $KL(I||H)$ is. Therefore, we mapped the inferred topic I_i to the

Table 4.3: The Contingency Table

		Gold judgments	
		YES	**NO**
Classifier judgments	**YES**	tp	fp
	NO	fn	tn

human topics H_j as $\underset{H_j}{argmin}\, KL(I_i||H_j)$.

4.2.2 Mapping Inferred Sentiments and Human Sentiments

We proposed a method to automatically guess the correspondence between the inferred and human sentiments. Intuitively, the top words with the high probability in the distribution of the inferred sentiment represent the meaning of the sentiment, i.e. positive, negative or neutral. We considered the sentiment scores of top M = 100 words with the highest probabilities in each inferred sentiment distribution S to calculate the sentiment value of S as in Equation (4.23).

$$sentiment(S) = \sum_{i=1}^{M} opinionValue(w_i) \tag{4.23}$$

where $opinionValue(w_i)$ is the opinion value of word w_i and calculated as in Equation (4.2).

The lowest, middle and highest sentiment scores of inferred sentiment distributions are mapped to "negative", "neutral" and "positive" sentiment categories, respectively.

4.3 Evaluation

4.3.1 Metrics

This subsection describes evaluation criteria for aspect-based sentiment analysis. Binary evaluation measures are calculated based on the number of true positives (tp), true negatives (tn), false positives (fp) and false negatives (fn). These can be summarized in the contingency table in Table 4.3.

For evaluation of binary classification, the accuracy is defined as Equation (4.24).

$$\textbf{Accuracy} = \frac{tp + tn}{tp + fp + fn + tn} \tag{4.24}$$

For evaluation of multiclass classification, let tp_c, fp_c, tn_c, fn_c be the number of true positives, false positives, true negatives and false negatives of a binary classifier for a label c. n_c denotes the number of instances in category c. C denotes a set of categories. Accuracy is the ratio of the number of correctly instances over by the total number of

Table 4.4: Statistics for the Aspect-Opinion Relation Dataset

Domain	# Products	# Sentences	# Relations
DVD Player	1	839	334
Cell Phone	2	1139	1043
Digital Camera	4	1497	1553
Diaper	1	375	227
MP3 Player	3	2725	1997

instances as Equation (4.25).

$$\textbf{Accuracy} = \frac{\sum_{c \in C} tp_c}{\sum_{c \in C} n_c} \tag{4.25}$$

Precision, Recall and F-measure are defined as macro average over individual categories as in Equation (4.26), (4.27) and (4.28). Note that the macro average is weighted by the number of instances in that category.

$$\textbf{Precision} = \frac{\sum_{c \in C} n_c \left(\frac{tp_c}{tp_c + fp_c} \right)}{|C|} \tag{4.26}$$

$$\textbf{Recall} = \frac{\sum_{c \in C} n_c \left(\frac{tp_c}{tp_c + fn_c} \right)}{|C|} \tag{4.27}$$

$$\textbf{F} = \frac{\sum_{c \in C} n_c \left(\frac{2 * tp_c}{2 * tp_c + fp_c + fn_c} \right)}{|C|} \tag{4.28}$$

4.3.2 Evaluation of Aspect-Opinion Relation Extraction

Dataset: We conducted experiments with labeled dataset developed by Wu et al. [94]. We also corrected some errors such as typing errors, aspect and opinion marking errors, and removed redundant relations. There are 5 domains (DVD Player, Cell Phone, Digital Camera, Diaper, MP3 Player) in this dataset. Table 4.4 shows the statistics of this dataset. In a given sentence, we consider aspect and opinion having no relation if it was not tagged in this sentence. Stanford CoreNLP [105] was used to parse constituent and dependency tree for each sentence.

Two experiments were designed. The first one is in-domain evaluation. This experiment tries to answer the question how well the models classify the data in the test set which is the same domain of the training data. We divided each domain into 80% for training and 20% for test. The second experiment is cross-domain evaluation. This evaluates the models on the test set which is different domain to the training data. We used

Table 4.5: In-domain Results of Aspect-Opinion Relation Extraction

Domain	Metric	SVM-B	CTK	DTK	CTK + DTK
DVD Player	A	0.804	**0.902**	0.863	**0.902**
	P	**0.905**	0.898	0.878	0.898
	R	0.864	**1.00**	0.977	**1.00**
	F	0.884	**0.946**	0.925	**0.946**
Cell Phone	A	0.728	0.712	**0.837**	0.815
	P	0.817	0.704	**0.884**	0.811
	R	0.764	**0.984**	0.870	0.943
	F	0.790	0.820	**0.877**	0.872
Digital Camera	A	0.721	0.652	**0.756**	0.709
	P	**0.798**	0.648	0.741	0.690
	R	0.746	**0.980**	0.940	0.975
	F	0.771	0.780	**0.829**	0.808
Diaper	A	**0.783**	0.739	0.739	0.739
	P	**0.929**	0.739	0.739	0.739
	R	0.765	**1.00**	**1.00**	**1.00**
	F	0.839	**0.850**	**0.850**	**0.850**
MP3 Player	A	0.800	0.705	0.800	**0.813**
	P	**0.923**	0.718	0.832	0.818
	R	0.769	**0.932**	0.885	**0.932**
	F	0.839	0.811	0.857	**0.872**

the sentences in "Digital Camera" and "Cell Phone" domains for training, and evaluated the models on "DVD Player", "Diaper" and "MP3 Player" domains. Accuracy, Precision, Recall and F-measure are used as the evaluation metrics. F-measure is the main metric to compare among four models SVM-B, CTK, DTK and CTK + DTK.

In-Domain Results: Table 4.5 summarizes the results of each domain in four metrics for each method. Figure 4.15 shows F-measure of four methods. SVM-B performed worst in all of the domains in F-measure. Our method CTK + DTK improves F-measure of SVM-B method by 6.2%, 8.2%, 3.7%, 1.1% and 3.3% in "DVD Player", "Cell Phone", "Digital Camera", "Diaper" and "MP3 Player", respectively. Therefore, our CTK + DTK method is better than SVM-B in in-domain evaluation. In addition, CTK + DTK method beats CTK in "Cell Phone", "Digital Camera" and "MP3 Player" domain and achieves competitive performance in "DVD Player" and "Diaper" domain. Thus, we can conclude that CTK + DTK is better than CTK method. Finally, CTK + DTK method is better than DTK in "DVD Player" and "MP3 Player" domain, comparable in "Cell Phone" and "Diaper" domain. To sum, DTK and CTK + DTK are the best methods for aspect-opinion relation extraction in in-domain evaluation.

Cross-Domain Results: The results of four methods in cross-domain are shown in Table 4.6. Figure 4.16 shows F-measure of four methods. Our method CTK + DTK outperformed the baseline SVM-B in all domains in F-measure. Improvements of 5.4%, 2.5% and 3.9% of F-measure are found in "DVD Player", "Diaper" and "MP3 Player"

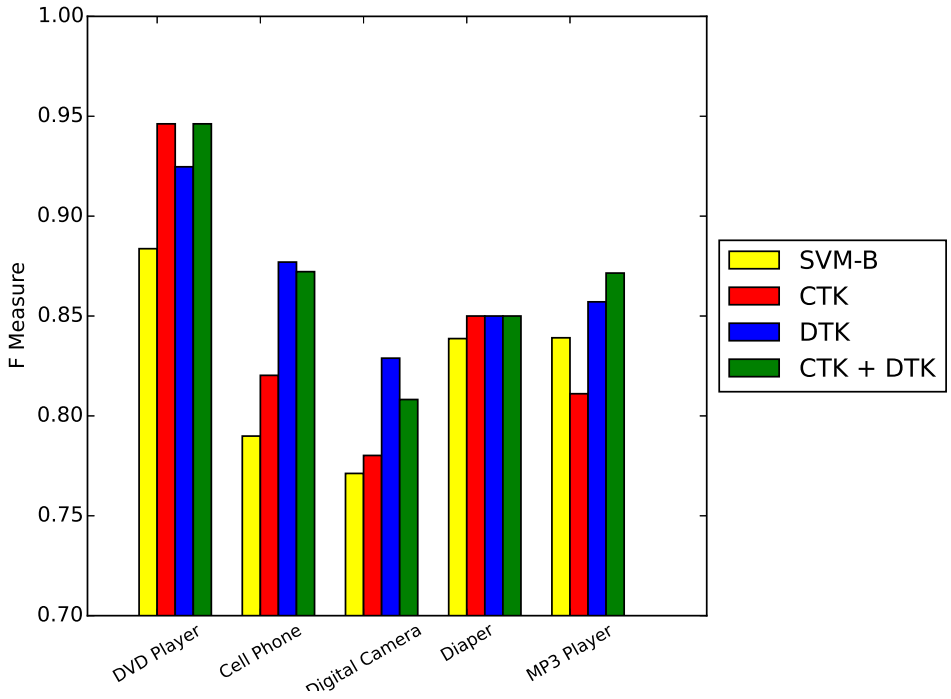

Figure 4.15: F-measure of Aspect-Opinion Relation Extraction Methods in In-Domain

domain, respectively. Therefore, our CTK + DTK method is better than SVM-B. In addition, CTK + DTK is better than CTK by 1.3%, 1.6% and 4.5% F-measure in each domain. Finally, compared with DTK, CTK + DTK shows 2.1% and 1.9% F-measure improvement in "DVD Player" and "Diaper" domain, and achieves competitive performance in "MP3 Player" domain. Therefore, we can conclude that CTK + DTK is the best method for extraction of aspect-opinion relations in the cross-domain evaluation.

4.3.3 Evaluation of Aspect Term Polarity Identification

Dataset: Because the data used in Subsection 4.3.2 is not annotated with the sentiment categories of the aspects, we used the restaurant reviews dataset in SemEval2014 Task 4 [3]. It consists of over 3000 English sentences of the restaurant reviews. For each sentence, the aspect terms and their polarity are annotated. The possible values of the polarity field are "positive", "negative", "neutral" and "conflict". Since, we do not deal with "conflict" category in our model, 84 sentences including the aspects with "conflict" polarity are removed from the dataset. CTK + DTK was trained from the sentences in "Digital Camera" and "Cell Phone" domains in Wu et al.'s dataset.

To investigate the effectiveness of integrating aspect-opinion relation extraction to aspect-based sentiment analysis, we compared the model with and without relation extraction (we call "ASA with RE" and "ASA w/o RE", respectively). Table 4.7 shows Accuracy, Precision, Recall and F-measure for all aspect phrases in the dataset. Preci-

[3]http://alt.qcri.org/semeval2014/task4/

Table 4.6: Cross-domain Results of Aspect-Opinion Relation Extraction

Domain	Metric	SVM-B	CTK	DTK	CTK + DTK
DVD Player	A	0.749	0.778	0.787	**0.808**
	P	**0.863**	0.793	0.859	0.834
	R	0.787	**0.952**	0.855	0.928
	F	0.824	0.865	0.857	**0.878**
Diaper	A	0.804	0.780	0.794	**0.812**
	P	**0.910**	0.786	0.846	0.823
	R	0.810	**0.964**	0.881	0.949
	F	0.857	0.866	0.863	**0.882**
MP3 Player	A	0.765	0.686	**0.792**	0.772
	P	**0.833**	0.683	0.805	0.760
	R	0.774	**0.954**	0.894	0.942
	F	0.802	0.796	**0.847**	0.841

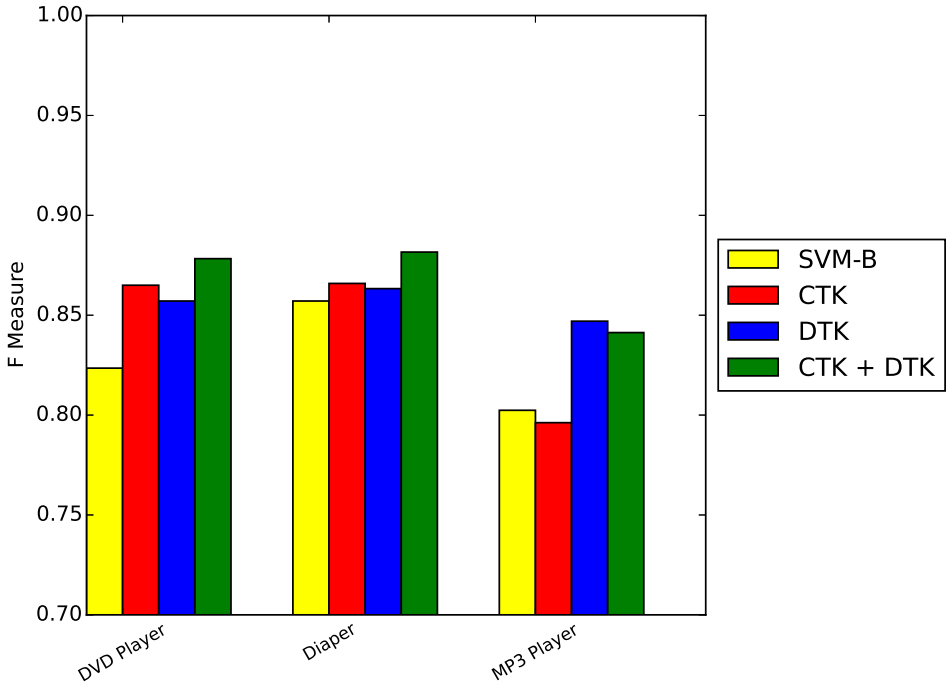

Figure 4.16: F-measure of Aspect-Opinion Relation Extraction Methods in Cross-Domain

Table 4.7: Results of Aspect Term Polarity Identification

Metric	ASA w/o RE	ASA with RE
A	0.465	**0.523**
P	**0.532**	**0.532**
R	0.465	**0.523**
F	0.477	**0.523**

Table 4.8: Results of Sentence-based for Aspect Term Polarity Identification

Metric	ASA w/o RE	ASA with RE
EMR	0.596	**0.634**
PMR	0.666	**0.702**

sion, Recall and F-measure are the average for three polarity categories weighted by the number of true instances. Accuracy of "ASA with RE" was 0.523. It outperformed the baseline by 5.8%. Furthermore, Recall and F-measure of "ASA with RE" were greatly improved. Table 4.8 shows the results of the sentence-based evaluation. Exact Match Ratio (EMR) is defined as a ratio of correctly classified sentences where the polarity of all aspects in the sentence are successfully identified. Partial Match Ratio (PMR) is the average of the partial matching scores of individual sentences, that is the proportion of the number of correctly classified aspects to all aspects in the sentence. "ASA with RE" was better 3.8% EMR and 3.6% PMR than "ASA w/o RE". From these results, we can conclude that using the aspect-opinion relation extraction is useful for sentiment analysis of aspects.

We use "ASA w/o RE", "ASA with RE", RNN and AdaRNN [60] as baseline models to compare with our PhraseRNN. To investigate the effectiveness of our PhraseRNN model, we will compare four configurations: PhraseRNN-1, PhraseRNN-2, PhraseRNN-3 and PhraseRNN-4 as explained in Subsection 4.1.5. We divided the restaurant review dataset into three parts with 70% training, 10% development and 20% test. Stanford CoreNLP [105] is used to parse the sentence and obtain constituent and dependency trees. For RNN, AdaRNN and PhraseRNN, the optimal parameters, which minimize the error in the development set, are used for the sentiment classification of the test set. We set $\beta = 1$ for AdaRNN and PhraseRNN since it is reported that $\beta = 1$ is the best parameter [60]. The optimized number of composition functions n and $m = \frac{n}{2}$ are selected by grid search with $n = \{2, 4, 6, 8, 10\}$ on the development set. $\lambda = 0.0001$ is employed. Table 4.9 shows the optimized parameters n for AdaRNN, PhraseRNN-3 and PhraseRNN-4.

Table 4.9: Optimized Parameter n for AdaRNN, PhraseRNN-3 and PhraseRNN-4

Methods	Optimized n
AdaRNN	2
PhraseRNN-3	10
PhraseRNN-4	4

Table 4.10: Results of Aspect Term Polarity Identification (cont.)

Methods	**A**	**P**	**R**	**F**
ASA w/o RE	0.4676	0.5463	0.4676	0.4806
ASA with RE	0.5239	0.5391	0.5239	0.5254
RNN	0.6085	0.5359	0.6085	0.5421
AdaRNN	0.6042	0.3678	0.6042	0.4573
PhraseRNN-1	0.6465†	0.5859†	0.6465†	0.5967*
PhraseRNN-2	0.6394†	**0.6240***	0.6394†	**0.6221***
PhraseRNN-3	**0.6620***	0.5388	**0.6620***	0.5932*
PhraseRNN-4	0.6592*	0.6026†	0.6592*	0.5980*

Notes: Statistical significance test of PhraseRNN comparing to RNN.
* Significant at the 1 percent level.
† Significant at the 5 percent level.

Results of aspect term polarity identification are shown in Table 4.10. The results indicate that four variations of our PhraseRNN outperform "ASA w/o RE", "ASA with RE", RNN and AdaRNN methods from 5.35% to 19.44% accuracy and 8% to 16.48% F-measure. Differences of PhraseRNN and RNN are verified by statistical significance tests. We use the paired randomization test because it does not require additional assumption about distribution of outputs [106]. As shown in Table 4.10, PhraseRNN significantly outperformed RNN in most cases. Among four variations, PhraseRNN-2 and PhraseRNN-3 achieved the best performance. By using different global functions in the inner and outer phrases, PhraseRNN-2 improves PhraseRNN-1 by 2.54% F-measure while keeping the comparable accuracy. Using multi-composition functions is also effective since PhraseRNN-3 was better than PhraseRNN-1 by 1.55% accuracy. PhraseRNN-4 improved PhraseRNN-3 by 6.38% precision while keeping comparable in other metrics.

Let us discuss the effectiveness of the integration of the dependency tree and the phrases in the constituent tree in PhraseRNN. Since our PhraseRNN-1 and PhraseRNN-3 outperform RNN and AdaRNN (the models relying on the binary dependency tree that is derived from the only dependency tree) respectively, we can conclude that our target dependent binary phrase dependency tree is much effective than binary dependency tree for aspect-based sentiment analysis.

In the data used in [60], one sentence contains only one aspect. On the other hand, two or more aspects can be appeared in one sentence in SemEval 2014 data. It is common in the real text. To examine in which cases our method is better than the others, we conduct an additional experiment by dividing the test set into three disjoint subsets. The first subset (**S1**) contains sentences having only one aspect. The second subset (**S2**) and third subset (**S3**) have two or more aspects in each sentence. All aspects in a sentence in S2 have the same sentiment category, while different sentiment categories in S3. The number of aspects in S1, S2 and S3 are 200, 323 and 187, respectively.

Table 4.11 shows the number of aspects where their sentiments are correctly identified by the methods in the subsets S1, S2 and S3. The accuracies are also shown in parentheses.

Table 4.11: The Number of Correctly Identified Aspects in Subsets S1, S2 and S3

Methods	S1	S2	S3
ASA w/o RE	98 (0.4900)	156 (0.4830)	78 (0.4171)
ASA with RE	111 (0.5550)	176 (0.5449)	85 (0.4545)
RNN	123 (0.6150)	226 (0.6997)	83 (0.4439)
AdaRNN	117 (0.5850)	234 (0.7245)	78 (0.4171)
PhraseRNN-1	**129** (**0.6450**)	248 (0.7678)	82 (0.4385)
PhraseRNN-2	125 (0.6250)	247 (0.7647)	82 (0.4385)
PhraseRNN-3	125 (0.6250)	**257** (**0.7957**)	88 (0.4706)
PhraseRNN-4	128 (0.6400)	250 (0.7740)	**90** (**0.4813**)

Table 4.12: Statistics for the Aspect Category Polarity Dataset

Domain	# Documents	# Aspect Categories
Digital Cameras	300	11
Laptops	300	15
Mobile Phones	300	15

Among three subsets, S3 is the most difficult and ambiguous case. In all methods, the performance in S3 is worse than S1 and S2. Comparing with other methods in each subset, PhraseRNN improves the accuracy in S2 more than in S1 and S3.

Accuracies of participating systems in SemEval 2014 Task 4 were between 0.4171 and 0.8095 [107]. However, these results cannot be simply compared to Table 4.7 and 4.10. Our method was evaluated on a training data of the task, while the participating systems were trained on it and evaluated on a separate test data. This test data is not available on the website of SemEval 2014 Task 4. In addition, as reported in the top system, DCU team [108], there is a big difference between training set and test set. The results for all of their systems in the test set is much higher (around 8.3%) than evaluated in 5-fold cross evaluation in the training set.

4.3.4 Evaluation of Aspect Category Polarity Identification

Dataset: We conducted the experiments with dataset from Lakkaraju et al. [68]. Table 4.12 shows the statistics of this dataset. The dataset contains 300 documents for each "Digital Cameras", "Laptops" and "Mobile Phones" domain. For each document, the authors have annotated the sentiment category ("positive", "neutral" and "negative") for each aspect category. There are 11, 15 and 15 aspect categories for these domains, respectively. Because the dataset is only annotated at the document level, we will conduct experiments to identify aspect categories and their sentiment categories for the documents rather than the sentences.

We run Collapsed Gibbs Sampling with 1000 iterations for training JST and TSLDA models. The number of hidden topics are set same as the number of aspect categories. By the algorithms described in Section 4.2, each hidden topic is mapped to one of the aspect categories.

Table 4.13: Results of Aspect Category Polarity Identification

Domain	Metric	JST	TSLDA
Digital Cameras	A	0.5579	**0.7085**
	P	0.6439	**0.6973**
	R	0.5579	**0.7085**
	F	0.595	**0.6775**
Laptops	A	0.7053	**0.8002**
	P	**0.7325**	0.712
	R	0.7053	**0.8002**
	F	0.7184	**0.7535**
Mobile Phones	A	0.6327	**0.742**
	P	0.6827	**0.71**
	R	0.6327	**0.742**
	F	0.6535	**0.6896**

Table 4.13 shows the accuracy, precision, recall and F-measure of the aspect-sentiment identification for three domains. Our model TSLDA outperformed JST by 15.06% Accuracy and 8.25% F-measure on "Digital Cameras" domain, 9.49% Accuracy and 3.51% F-measure on "Laptops" domain, 10.93% Accuracy and 3.61% F-measure on "Mobile Phones" domain. Therefore, we can conclude that our TSLDA model is better than JST model for aspect category polarity identification.

4.4 Summary

We applied two kernels of constituent and dependency trees and proposed the new tree kernel for aspect-opinion relation extraction. The results showed that the models using tree kernels outperformed the baseline SVM-B. Our tree kernel based model for aspect-opinion relation extraction can be further improved by using semantic information from semantic trees. Combining the syntactic tree and semantic tree for calculating tree kernel will be explored in our future work.

Furthermore, we proposed the new method for identifying the sentiment categories of the aspect terms in the sentences with the relation extraction module. Our method achieved better performance in almost all metrics compared to the method without relation extraction. The "ASA with RE" could be further improved by learning the weight parameters in Equation (4.4) from the training data. The automatically learnt weights could capture more accurately how the aspect-opinion relations contribute to the sentiments of the aspects.

A new supervised method, PhraseRNN, was proposed to identify the sentiment category of an aspect in a sentence. By combining the constituent and dependency tree, the model firstly converts it to a target dependent binary phrase dependency tree. Then, it constructs the aspect representation by recursively combining child nodes through the new tree structure. To predict the sentiment category for the aspect, the derived aspect

representation vector was inputted to a logistic regression. The results indicate that our PhraseRNN is much better than "ASA w/o RE", "ASA with RE", RNN and AdaRNN methods for aspect-based sentiment analysis.

For the aspect category polarity identification, we proposed a new method for automatically mapping the inferred aspect and sentiment to gold aspect and sentiment in two topic models JST and TSLDA. We also empirically evaluated two topic models. We found that our TSLDA achieved better performance than JST.

Chapter 5

Sentiment Analysis for Stock Prediction

In this chapter, we discuss sentiment analysis models on financial domains and use these models to predict the future stock prices. Basically, in this thesis, the information sources for the stock prediction are the past prices of the stock and the text about the company of it. First, in Section 5.1, the dataset used for the prediction of the stock price or its movement will be shown. Section 5.2 describes the methods to predict the stock prices, while 5.3 proposes the methods to predict the stock price movement. Section 5.4 reports the experiments to evaluate the proposed methods. Section 5.5 presents the summary of this chapter.

5.1 Dataset

We used two datasets for our stock prediction model. The first one is the historical price dataset, and the second one is the mood information dataset.

5.1.1 Historical Prices

Historical prices are extracted from Yahoo Finance for the 18 stocks. The list of stock quotes and company names is shown in Table 5.1. For each transaction date, there are open, high, low, close and adjusted close prices. The adjusted close prices are the close prices which are adjusted for dividends and splits. Figure 5.1 shows these prices (as well as the volumes) for a one month period from August 1 to 31, 2012 of stock YHOO. The adjusted close price is often used for stock market prediction in other researches [84]. Therefore, we chose it as the stock price value for each transaction date.

5.1.2 Message Board Dataset

To get the mood information of the stocks, we collected the 18 message boards of 18 stocks from Yahoo Finance Message Board for a period of one year (from July 23, 2012 to

Date	Open	High	Low	Close	Volume	Adj Close*
Aug 31, 2012	14.79	14.82	14.59	14.65	11,619,700	14.65
Aug 30, 2012	14.81	14.84	14.64	14.67	10,698,800	14.67
Aug 29, 2012	14.73	14.94	14.70	14.84	21,113,600	14.84
Aug 28, 2012	14.84	14.87	14.69	14.72	12,706,400	14.72
Aug 27, 2012	14.92	14.93	14.77	14.85	10,054,000	14.85
Aug 24, 2012	14.82	14.94	14.77	14.92	8,650,400	14.92
Aug 23, 2012	14.90	14.97	14.82	14.87	12,463,000	14.87
Aug 22, 2012	14.95	14.99	14.86	14.92	9,168,400	14.92
Aug 21, 2012	14.95	15.01	14.88	14.97	27,934,700	14.97
Aug 20, 2012	14.99	15.05	14.88	14.96	11,193,900	14.96
Aug 17, 2012	15.02	15.07	14.85	15.03	19,640,700	15.03
Aug 16, 2012	14.81	15.01	14.75	14.99	24,971,900	14.99
Aug 15, 2012	14.77	14.86	14.65	14.76	20,682,900	14.76
Aug 14, 2012	15.04	15.05	14.69	14.73	29,655,200	14.73
Aug 13, 2012	15.03	15.21	15.00	15.02	20,849,400	15.02
Aug 10, 2012	15.25	15.35	15.01	15.15	61,987,300	15.15
Aug 9, 2012	16.16	16.16	15.98	16.01	8,613,100	16.01
Aug 8, 2012	16.15	16.32	16.09	16.17	7,379,000	16.17
Aug 7, 2012	16.09	16.37	16.07	16.22	17,281,700	16.22
Aug 6, 2012	16.00	16.07	15.95	16.04	8,803,900	16.04
Aug 3, 2012	15.89	16.03	15.82	15.97	9,140,800	15.97
Aug 2, 2012	15.86	16.00	15.64	15.75	12,900,500	15.75
Aug 1, 2012	15.86	16.07	15.83	15.99	14,008,000	15.99

Figure 5.1: YHOO Historical Prices

Table 5.1: Quotes and Company Names

Stocks	**Company Names**
AAPL	Apple Inc.
AMZN	Amazon.com Inc.
BA	The Boeing Company
BAC	Bank of America Corporation
CSCO	Cisco Systems Inc.
DELL	Dell Inc.
EBAY	eBay Inc.
ETFC	E Trade Financial Corporation
GOOG	Google Inc.
IBM	International Business Machines Corporation
INTC	Intel Corporation
KO	The Coca-Cola Company
MSFT	Microsoft Corporation
NVDA	NVIDIA Corporation
ORCL	Oracle Corporation
T	AT&T Inc.
XOM	Exxon Mobil Corporation
YHOO	Yahoo! Inc.

July 19, 2013) [1]. On the message boards, users usually discuss company news, prediction about stock going up or down, facts, comments (usually negative) about specific company executives or company events. In 15.6% messages of this dataset, when users posted messages on the message boards, they annotated each message as one of the following sentiment tags: Strong Buy, Buy, Hold, Sell and Strong Sell. There are two kinds of messages. The first one is the messages created by starting a new topic. The other is reply messages to existing messages. Most of users' posts are reply messages. Interaction between the users makes a complicated communication network. In our research, however, we treated all messages are independent from each other.

Figure 5.2 shows an example message from AAPL Message Board. In this message, on July 6, 2012, a username "keepshorting" posted the message "Looks like the competition is heating up. $199 tablet, what is next? $999 laptops and then $499 laptops? the margins are impossible to keep up. impossible folks." to reply to a message of another user. In addition, this user selected the sentiment for this stock as "Strong Sell".

The stock market is not opened at the weekend and holiday. To assign the messages to the transaction dates, the messages which were posted from 4 pm of the previous transaction date to 4 pm of the current transaction date will belong to the current transaction. We choose 4 pm because it is the time of closing transaction. There are 249 transaction dates in one year in the dataset. Table 5.2 summarizes the statistics of our dataset for each transaction date: the min, median, mean and max of the number of messages and

[1]The AAPL message board has the highest number of messages. Because of the limitation on the number of web pages, we can only collect for a period of seven months for this stock.

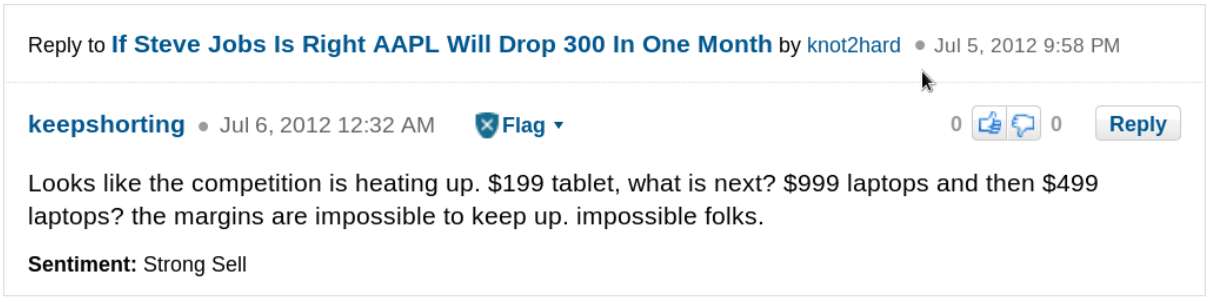

Figure 5.2: A Message from AAPL Message Board

Table 5.2: Statistics of Our Dataset for Each Transaction Date

Stocks	The Number of Messages				Mean of the Number of Human Sentiments
	Min	Median	Mean	Max	
AAPL	0	1093	1678	11220	350
AMZN	24	154	192	1963	28
BA	46	173	203	1053	16
BAC	94	282	343	1366	49
CSCO	69	247	274	972	10
DELL	0	18	42	587	10
EBAY	1	17	29	267	3
ETFC	2	42	56	315	12
GOOG	10	69	93	1305	16
IBM	3	14	20	195	3
INTC	37	177	200	958	29
KO	0	6	8	89	2
MSFT	27	139	172	815	53
NVDA	10	65	80	410	11
ORCL	5	67	79	372	6
T	10	52	59	251	8
XOM	10	37	44	202	4
YHOO	22	121	141	860	27

the mean of the number of existing sentiments annotated by users.

Some previous work used Twitter as the mood information source for sentiment analysis related to a particular stock. There are some reasons why in our research Twitter is not chosen as a mood source. The first one is the information in Twitter seems to be messier than that in the message board. In Twitter, users discuss many things. Even though, tweets can be filtered by some rules such as using hashtag (#AAPL, $AAPL and so on) to find relevant tweets, the lack of consistency among posters in hashtag use and the existence of a large amount of noisy posts makes finding post related to a specific stock difficult. The second reason is the way to collect tweets. There are two ways to collect tweets from Twitter. The first one is from the Twittter Searching API. This only allows to search tweets from one week in the past for free. The other way is using Twitter Streaming API. It allows to collect the real time tweets rather than search from the history.

Table 5.3: Features of the Stock Prediction Model

Method	**Features**
Price Only	$price_{t-1}$, $price_{t-2}$
Human Sentiment	$price_{t-1}$, $price_{t-2}$, $Hsent_{i,t}$, $Hsent_{i,t-1}$
Sentiment Classification	$price_{t-1}$, $price_{t-2}$, $Csent_{i,t}$, $Csent_{i,t-1}$

However, to collect tweets in one year period, it takes one year. Those make difficult to gather data from Twitter. Finally, there are no explicit sentiment annotated by posters in Twitter. There is no way to compare between human sentiment and automatically identified sentiment.

However, as in other mood information sources, the messages on the message board are also messy. The text is usually short, contains many misspellings, uncommon grammar constructions and so on. Moreover, the false and unrelated information also exists.

5.2 Methods for Stock Prediction

In this section, we aim at predicting the stock value in future. Linear Regression and Support Vector Regression, which are introduced in Subsection 2.1.4 and 2.1.5, are used as the prediction models. To assess the effectiveness of sentiment analysis on the message boards, three sets of features are designed. The first one used only the historical prices. The other methods incorporated the mood information into the prediction model. Table 5.3 summarizes our features used in the model to predict the exact price value at the transaction date t. The details of each feature will be explained in the next subsections.

5.2.1 Price Only

In this method, only historical prices are used to predict the stock movement. This model is implemented to investigate whether there are patterns in the history of the stock or not. In addition, it is used as a baseline to evaluate whether integration of the sentiments is effective by comparing with other sentiment models. Features used for the training are $price_{t-1}$ and $price_{t-2}$ which are the price values at the transaction dates $t-1$ and $t-2$, respectively.

5.2.2 Human Sentiment

In addition to historical prices, this model integrates the sentiments annotated by human into the prediction model. As denoted in Subsection 5.1.2, in 15.6% of the posts in the MessageBoard dataset, the users explicitly select a sentiment label with their posts. These sentiment labels are "Strong Buy", "Buy", "Hold", "Sell" and "Strong Sell". Instead of using all the messages, we try to use only messages with annotated sentiments by users, and discard the other messages. From these messages, we use only the explicit sentiment and remove other information such as message content. This model is implemented to

explore how mood annotated by human can be used to predict the stock. Because the sentiments are annotated by human, this feature may be one of the most effective features for stock prediction.

For each transaction date t, the percentage of each class (Strong Buy, Buy, Hold, Sell, Strong Sell) is calculated. The percentage of a class is the number of messages annotated with the sentiment class to the total number of messages at the current transaction date t. Then, we integrate them into the prediction model.

Features used for training model are $price_{t-1}$, $price_{t-2}$, $Hsent_{i,t}$ and $Hsent_{i,t-1}$. $Hsent_{i,t}$ and $Hsent_{i,t-1}$ are the percentages of the number of messages belonging to the sentiment class i ($i \in$ {Strong Buy, Buy, Hold, Sell, Strong Sell}) at the transaction dates t and $t-1$, respectively.

5.2.3 Sentiment Classification

One of the disadvantage of the human sentiment model is that it utilizes the sentiments of the only small amount (15.6%) of the messages. To utilize the remaining 84.4% of the messages without the explicit sentiments, we try to build a model to extract the sentiments for those messages. A classification model is trained from messages with annotated sentiments on the training dataset. Then this classification model is used to classify the remaining messages into five classes (Strong Buy, Buy, Hold, Sell, Strong Sell).

SVM with the linear kernel is used as the classification model. The features used for training SVM is bag-of-words in the title and content of the messages. To extract bag-of-words features, first we remove the stop words from messages. Then, all the words are lemmatized by the Stanford CoreNLP [105]. Feature weighting is a method assigning appropriate weights to the features in order to reflect how important the features are in the messages. TF-IDF is used as the weight of the feature in this model. It is a combination of **Term Frequency** and **Inverse Document Frequency**. TF-IDF of $word_i$ in $document_j$ is:

$$w_{ij} = tf_{ij} * (log_{10}\frac{M}{DF_i} + 1) \tag{5.1}$$

tf_{ij}: frequency of $term_i$ in $document_j$.
DF_i: document frequency of $term_i$.
M: the total number of documents in a corpus.

As in Human Sentiment feature, we also calculated the percentage of the number of messages of each class for each transaction date. Features used for training the model are $price_{t-1}$, $price_{t-2}$, $Csent_{i,t}$ and $Csent_{i,t-1}$. $Csent_{i,t}$ and $Csent_{i,t-1}$ are similar to $Hsent_{i,t}$ and $Hsent_{i,t-1}$, but both messages with human annotated sentiment and automatically classified sentiments are used to calculate the percentages of the number of messages belonging to the sentiment class i.

Table 5.4: Features of the Stock Movement Prediction Model

Method	**Features**
Price Only	$price_{t-1}$, $price_{t-2}$
Human Sentiment	$price_{t-1}$, $price_{t-2}$, $Hsent_{i,t}$, $Hsent_{i,t-1}$
Sentiment Classification	$price_{t-1}$, $price_{t-2}$, $Csent_{i,t}$, $Csent_{i,t-1}$
LDA-based Method	$price_{t-1}$, $price_{t-2}$, $lda_{i,t}$, $lda_{i,t-1}$
JST-based Method	$price_{t-1}$, $price_{t-2}$, $jst_{i,j,t}$, $jst_{i,j,t-1}$
TSLDA-based Method	$price_{t-1}$, $price_{t-2}$, $tslda_{i,j,t}$, $tslda_{i,j,t-1}$
Aspect-based Sentiment	$price_{t-1}$, $price_{t-2}$, $Asent_{i,t}$, $Asent_{i,t-1}$, $I_{i,t}$, $I_{i,t-1}$

5.3 Methods for Stock Movement Prediction

In this section, we aim at predicting the price movement, that is, if the stock value is up or down. Support Vector Machine (SVM) has long been recognized as being able to efficiently handle high dimensional data and has been shown to perform well on classification [101, 102]. Therefore, we chose SVM with the linear kernel as the prediction model. To assess the effectiveness of sentiment analysis on these message boards, seven sets of features are designed. The first one used only the historical prices. The other methods incorporate the mood information into the prediction model. All the feature values are scaled into the $[-1, 1]$ value. Table 5.4 summarizes our features used in the model to predict the price movement at the transaction date t. The details of each feature will be explained in the next subsections.

5.3.1 Price Only

In this method, only historical prices are used to predict the stock movement. This model is considered to investigate whether there are patterns in the history of the stock or not. In addition, this model is used as a baseline to evaluate whether the sentiments are effective by comparing with other sentiment models. Features used for training SVM are $price_{t-1}$ and $price_{t-2}$ which are the price movements (up, down) at the transaction dates $t-1$ and $t-2$, respectively.

5.3.2 Human Sentiment

We integrate the sentiment annotated by human into the prediction model as in Subsection 5.2.2. Features used for training SVM are $price_{t-1}$, $price_{t-2}$, $Hsent_{i,t}$ and $Hsent_{i,t-1}$. $Hsent_{i,t}$ and $Hsent_{i,t-1}$ are the percentages of the number of messages belonging to the sentiment class i ($i \in$ {Strong Buy, Buy, Hold, Sell, Strong Sell}) at the transaction dates t and $t-1$, respectively.

5.3.3 Sentiment Classification

We use the sentiments of all messages in the message board as in Subsection 5.2.3. Features used for training SVM are $price_{t-1}$, $price_{t-2}$, $Csent_{i,t}$ and $Csent_{i,t-1}$. $Csent_{i,t}$ and $Csent_{i,t-1}$ are similar to $Hsent_{i,t}$ and $Hsent_{i,t-1}$, but both messages with human annotated sentiment and automatically classified sentiments are used to calculate the percentages of the number of messages belonging to the sentiment class i.

5.3.4 LDA-based Method

In this model, we consider each message as a mixture of hidden topics. As described in Section 3.2, Latent Dirichlet Allocation (LDA) [88] is a generative probabilistic model of a corpus. The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. Therefore, we choose LDA as a simple topic model to discover these hidden topics [2].

We removed the stop words from messages. Then, all the words are lemmatized by the Stanford CoreNLP. We train LDA on the training set, and infer the topics for unseen messages on the test set. Topics are inferred by Gibbs Sampling with 1000 iterations. After that, the probability of each topic for each message is calculated. Next, for each transaction date t, the probability of each topic is defined as the average of the probabilities of that topic in the messages belonging to that transaction date. Then we integrate these probabilities into the prediction model.

Features used for the training SVM are $price_{t-1}$, $price_{t-2}$, $lda_{i,t}$ and $lda_{i,t-1}$. $lda_{i,t}$ and $lda_{i,t-1}$ are the probabilities of the topic i ($i \in \{1, \cdots, K\}$) for the transaction dates t and $t-1$. The number of the topics K is determined as explained in Section 5.4.

5.3.5 JST-based Method

The opinion is often expressed on a topic or aspect. When people post the message on the social media to express their opinion for a given stock, they tend to talk their opinions for a given topic or aspect such as profit and dividend. Based on pairs of topic-sentiment, they would think that the future price of that stock goes up or down. From that intuition, we propose a new feature topic-sentiment for the stock prediction model. To extract pairs of topic-sentiment, we tried to use three kinds of models. The first two models are latent topic based models, JST model [66] and TSLDA. The other is Aspect-based Sentiment model which will be discussed in Subsection 5.3.7. This subsection describes the first one, the JST-based model.

We consider each message as a mixture of hidden topics and sentiments. JST was used to extract topics and sentiments simultaneously. After removal of stop words and lemmatization, the JST model is trained from the training set, and topics on the test set are inferred by Gibbs Sampling with 1000 iterations. We chose 3 as the number of sentiments which might represent negative, neutral and positive. The number of the

[2]We used the LDA implementation from the Mallet library.

topics K is determined as explained in Section 5.4. Next, the joint probability of each pair of topic and sentiment is calculated for each message. After that, for each transaction date t, the joint probability of each topic-sentiment pair is defined as the average of the joint probabilities of that in the messages belonging to that transaction date. Then we integrate these probabilities into the prediction model.

Features used for the training SVM are $price_{t-1}$, $price_{t-2}$, $jst_{i,j,t}$ and $jst_{i,j,t-1}$. $jst_{i,j,t}$ and $jst_{i,j,t-1}$ are the joint probabilities of the sentiment i ($i \in \{1, 2, 3\}$) and topic j ($j \in \{1, \cdots, K\}$) for the transaction dates t and $t-1$.

5.3.6 TSLDA-based Method

We use our TSLDA model described in Chapter 3 to capture the topics and sentiments simultaneously. First, a rule-based algorithm is applied to identify the category of each word in the documents. Consecutive nouns are considered as topic words. If a word is not a noun and in a list of opinion words in SentiWordNet [45], it is considered as an opinion word. The rest of words are classified as background words.

After lemmatization, TSLDA model is trained by Collapsed Gibbs Sampling with 1000 iterations. We chose 3 as the number of sentiments which represent for negative, neutral and positive. K (number of topics) is determined as explained in Section 5.4. The topic and its sentiment in each sentence are gotten from the topic assignment and sentiment assignment in TSLDA. If there is a sentence expressing the sentiment j on the topic i, we represent the tuple $(i, j) = 1$, and 0 otherwise. The proportion of (i, j) over all sentences are calculated for each message. For each transaction date, a weight of the tuple (i, j) is defined as the average of the proportions over all messages. Then we integrated the weights of the topics and their sentiments into the prediction model.

Features used for training SVM are $price_{t-1}$, $price_{t-2}$, $tslda_{i,j,t}$ and $tslda_{i,j,t-1}$. $tslda_{i,j,t}$ and $tslda_{i,j,t-1}$ are the weights of the topic i ($i \in \{1, \cdots, K\}$) with the sentiment j ($j \in \{1, 2, 3\}$) for the transaction dates t and $t-1$.

5.3.7 Aspect-based Sentiment

Instead of considering the mixtures of hidden topics and sentiments as in the previous models, this model considers explicit topics and sentiments in the text. Each message is represented as a list of topics and their corresponding sentiment values. The topic is defined as the consecutive nouns in the sentence. For example, the message "The profit will go up." contains the topic "profit" and a positive sentiment "up" for that topic as in Figure 5.3.

We propose a new model to calculate the sentiment values of topics in a sentence. Each message are split into the sentences, then the Stanford CoreNLP is used for POS tagging and lemmatization of them. First, we extract the topics in the training dataset by the algorithm shown in Figure 5.4. We consider the consecutive nouns as the topic in the sentence. To eliminate rare topics, topics occurring less than 10 are removed from the list

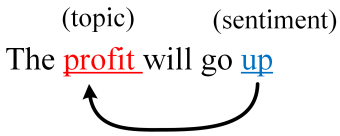

Figure 5.3: An Example Sentence with Topic and Its Sentiment

Input: Training dataset
Output: List of topics of this dataset
1 Extract consecutive noun words in each sentence as a topic ;
2 Remove topics that appear less than 10 times in the training dataset ;

Figure 5.4: Algorithm for Extracting Topics from Dataset

of the topics. Next, we extract the sentiment value of each topic in the list in each sentence by the algorithm shown in Figure 5.5. For each sentence, opinion words are identified based on the list of opinions from SentiWordNet [45]. SentiWordNet is a lexical resource for opinion mining. SentiWordNet assigns three sentiment scores: positivity, objectivity and negativity for each word. We combine the scores of positivity and negativity into a single opinion value as line 7 in Figure 5.5. The closer the distance between the topic phrase and the opinion word is, the more highly the opinion may associate with the topic. Therefore, the sentiment value of a topic phrase in a sentence is the summation over all opinion values divided by their distance to that topic as line 8 in Figure 5.5.

For each message, the sentiment value of each topic is defined as the average of the sentiment scores of that topic in the sentences. Finally, for each transaction date t, the sentiment value for each topic is the average of the sentiment values of that topic in the messages belonging to that transaction date.

In addition to the sentiment values of the topics, the importance of topics for each transaction date is also considered. Intuitively, some topics have more impact on the prediction than others. If a topic is discussed in many messages, it might be an important topic for the given transaction date. The importance of a topic i in a transaction date t is calculated as in Equation (5.2).

$$I_{i;t} = \frac{N_{i;t}}{N_t} \tag{5.2}$$

where

$I_{i;t}$: the importance of topic i in the transaction date t.

$N_{i;t}$: the number of messages containing the topic i in the transaction date t.

N_t: the number of messages in the transaction date t.

That is, $I_{i;t}$ is defined as the ratio of the number of messages containing the topic i in the transaction date t to the total number of the messages in t.

The sentiment scores of the topics at the transaction date and their importance are used in the prediction model. Features used for the training SVM are $price_{t-1}$, $price_{t-2}$, $Asent_{i;t}$, $Asent_{i,t-1}$, $I_{i;t}$ and $I_{i;t-1}$. $Asent_{i;t}$ and $Asent_{i,t-1}$ are the sentiment values of the

Input: A sentence
Output: List of pairs $(topic, sentimentValue)$ for this sentence
Extract topics in this sentence (Based on the list of topics extracted from the algorithm shown in Figure 5.4) ;
Extract opinion words in the sentence by using SentiWordNet ;
for *each topic* t_i *in the sentence* **do**
    for *each opinion* o_j *in this sentence* **do**
        Calculate $distance(t_i, o_j)$ = position distance between topic t_i and opinion word o_j ;
        Get pos_score, neg_score of opinion o_j from SentiWordNet ;
        Calculate $opinionValues(o_j) = \frac{pos\_score - neg\_score}{pos\_score + neg\_score}$;
        $sentimentValue_{t_i} = sentimentValue_{t_i} + \frac{opinionValues(o_j)}{distance(t_i, o_j)}$;
        Add $(t_i, sentimentValue_{t_i})$ to the list of pairs $(topic, sentimentValue)$
    end
end

Figure 5.5: Algorithm for Extracting Topics and Their Sentiment Values

topic i at the transaction dates t and $t-1$. While, $I_{i;t}$ and $I_{i;t-1}$ are the importance of the topic i at the transaction dates t and $t-1$.

5.4 Evaluation

5.4.1 Evaluation of Sentiment Classification

As discussed in Subsection 5.2.3 and 5.3.3, the sentiment classification method classifies the unannotated messages into one of five sentiment classes using the annotated messages as training instances. To evaluate how this model can correctly classify the sentiments of the messages, we divided the annotated sentiments into two subsets: 70% for training and 30% for testing. We used Accuracy, Precision, Recall and F-measure as the evaluation metrics. Precision, Recall and F-measure are the average for sentiment categories weighted by the number of true instances. Table 5.5 shows the result of this classification. The last row shows the average of the 18 stocks. On average accuracy of 18 stocks, the model can classify the messages into 5 sentiment categories with 64.04% accuracy and 59.17% F-measure.

5.4.2 Evaluation of Stock Prediction

Experiment Setup

The performance is evaluated using Mean Absolute Error (MAE) and Directional Accuracy metric. MAE is a quantity to measure how predicted prices are close to the actual prices. Equation(5.3) shows the definition of MAE. The smaller value of MAE is, the

Table 5.5: Results of Sentiment Classification

Stock	**A**	**P**	**R**	**F**
AAPL	0.6591	0.6744	0.6591	0.5834
AMZN	0.7042	0.6884	0.7042	0.6506
BA	0.702	0.666	0.702	0.6716
BAC	0.6158	0.5963	0.6158	0.5659
CSCO	0.5391	0.5647	0.5391	0.4904
DELL	0.6015	0.6025	0.6015	0.5724
EBAY	0.6884	0.6229	0.6884	0.6401
ETFC	0.75	0.7686	0.75	0.7201
GOOG	0.5472	0.5232	0.5472	0.5009
IBM	0.5732	0.7217	0.5732	0.4827
INTC	0.7185	0.7138	0.7185	0.6605
KO	0.5089	0.4931	0.5089	0.4692
MSFT	0.6988	0.7011	0.6988	0.6751
NVDA	0.6971	0.7207	0.6971	0.666
ORCL	0.6226	0.5761	0.6226	0.5763
T	0.6201	0.69	0.6201	0.5783
XOM	0.5727	0.5065	0.5727	0.5283
YHOO	0.7084	0.7052	0.7084	0.6192
AVERAGE	0.6404	0.6409	0.6404	0.5917

better the model is.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \mid y_i - t_i \mid \tag{5.3}$$

where

N: the number of transactions in the test set.

t_i: the true price at transaction i.

y_i: the predicted price at transaction i.

When the stock value can be guessed, the stock movement (up or down) can be also predicted. Therefore, the performance on the prediction of the stock movement is also evaluated in this experiment. Directional accuracy measures how accurate the model can predict if the price moves up or down. The predicted movement is assigned by comparing the predicted price with the true price at the previous transaction date. The true movement is assigned by comparing the true price of the current and previous date. Directional accuracy is defined as Equation (5.4).

$$DirectionalAccuracy = \frac{tp + tn}{tp + fp + fn + tn} \tag{5.4}$$

where

tp: the number of samples correctly categorized for positive samples.

tn: the number of samples correctly rejected for the negative samples.

fp: the number of samples incorrectly categorized for the positive samples.

Table 5.6: Results of MAEs of 18 Stocks Using Linear Regression

Stocks	Price Only	Human Sentiment	Sentiment Classification
AAPL	10.0099	7.21036381136e+14	4.29551824292e+14
AMZN	3.4398	3.7851	3.9667
BA	1.3095	1.3053	1.3462
BAC	0.1736	0.1783	0.1602
CSCO	0.2568	0.2488	0.2807
DELL	0.0732	0.1104	0.0945
EBAY	0.7292	0.792	0.7313
ETFC	0.1867	0.2071	0.1877
GOOG	7.0791	5.43542175206e+13	6.8349
IBM	1.673	1.6416	1.6553
INTC	0.2768	0.3838	0.2783
KO	0.3628	0.3931	0.3789
MSFT	0.4345	0.4605	0.4105
NVDA	0.1428	0.2256	0.1422
ORCL	0.4145	0.4027	0.4356
T	0.3024	0.3073	0.3416
XOM	0.7203	0.8555	0.7778
YHOO	0.4422	0.4413	0.4434
AVERAGE	**1.5571**	4.30772554809e+13	2.38639902385e+13

fn: the number of samples incorrectly rejected for the negative samples.
In this experiment, the positive and negative samples stand for the transaction dates where the price is moved up and down, respectively.

We divided the time series into two parts: the period from July 23, 2012 to March 28, 2013 for training containing 171 transaction dates, and April 01, 2013 to July 19, 2013 for testing containing 78 transaction dates [3].

Using Linear Regression as the Prediction Model

The linear regression is used as the prediction model. The results of MAE are shown in Table 5.6. In addition to the result of each stock, we also calculated the average of 18 stocks for each model. However, MAE are very different for 18 stocks, comparison of the average MAE is not meaningful.

Let us compare the number of stocks for which MAE is higher or lower for comparison between two methods. Compared with "Price Only" method, "Human Sentiment" method has lower MAE in 7 stocks and higher MAE in 11 stocks. The "Sentiment Classification" method has lower MAE in 6 stocks and higher MAE in 12 stocks. Therefore, integrating sentiments seems to be not helpful to predict the exact price in linear re-

[3] As explained in Subsection 5.1.2, the collected Message Board dataset of AAPL has only seven months unlike one year period of other stocks. For AAPL, the period July 06, 2012 - October 01, 2012 (61 transactions) is used for training, and November 12, 2012 to March 13, 2013 (83 transaction dates) is used for test.

Table 5.7: Results of Directional Accuracies of 18 Stocks Using Linear Regression

Stocks	Price Only	Human Sentiment	Sentiment Classification
AAPL	0.3846	0.4103	0.5128
AMZN	0.3902	0.4634	0.4634
BA	0.5854	0.5122	0.439
BAC	0.5122	0.5366	0.5366
CSCO	0.4634	0.439	0.4634
DELL	0.3415	0.3415	0.4146
EBAY	0.4878	0.4146	0.561
ETFC	0.3659	0.3902	0.4634
GOOG	0.439	0.561	0.4878
IBM	0.5854	0.6829	0.6098
INTC	0.5122	0.5122	0.5366
KO	0.4878	0.5122	0.4878
MSFT	0.5366	0.5366	0.4878
NVDA	0.561	0.561	0.6585
ORCL	0.561	0.561	0.4634
T	0.5122	0.561	0.4634
XOM	0.4146	0.4878	0.439
YHOO	0.4634	0.4634	0.439
AVERAGE	0.478	**0.497**	0.496

gression. Comparing with "Sentiment Classification", "Human Sentiment" method lower MAE in 8 and higher in 10 stocks. The use of only a small amount of human annotated sentiments of the messages is worse than using a large amount of automatically identified sentiments.

The results of directional accuracy are shown in Table 5.7. In addition to the result of each stock, we also calculated the average of 18 stocks for each model for easy comparison.

Even though integrating sentiments could not help to improve predicting the exact price in linear regression, the directional accuracies are better when we integrate sentiments. The average directional accuracy is improved by 1.9% when integrating human sentiments, and 1.8% when using sentiment classification.

Using Support Vector Regression as the Prediction Model

Support Vector Regression is used as the prediction model. The results of MAE are shown in Table 5.8. In addition to the result of each stock, we also calculated the average of 18 stocks for each model for easy comparison.

The average MAEs of 18 stocks in "Human Sentiment" is lower than in "Price Only" method. In addition to average MAE of 18 stocks, we also compare the methods by counting the number of stocks where MAE of one method is higher or lower than the other. Comparing with "Price Only" method, "Human Sentiment" method has lower MAE in 3 stocks and higher MAE in 15 stocks. The "Sentiment Classification" method

Table 5.8: Results of MAEs of 18 Stocks Using SVR

Stocks	Price Only	Human Sentiment	Sentiment Classification
AAPL	60.0171	50.6098	55.0351
AMZN	4.8313	5.8163	5.8307
BA	1.2884	1.4931	1.4069
BAC	0.1809	0.1886	0.1722
CSCO	0.2538	0.2692	0.2713
DELL	0.0703	0.0898	0.0988
EBAY	0.738	0.7979	0.7796
ETFC	0.1856	0.2004	0.2
GOOG	24.7702	23.6819	28.3316
IBM	1.5933	1.6956	1.674
INTC	0.2936	0.4061	0.2821
KO	0.3608	0.3952	0.3775
MSFT	0.4538	0.4717	0.5086
NVDA	0.1443	0.2269	0.1555
ORCL	0.4226	0.4062	0.4124
T	0.2943	0.2987	0.3236
XOM	0.6843	0.9115	0.8163
YHOO	0.4506	0.4558	0.4618
AVERAGE	5.3907	**4.9119**	5.3966

has lower MAE in 5 stocks and higher MAE in 13 stocks. However, comparing with "Sentiment Classification", "Human Sentiment" method lower MAE in 9 and higher in 9 stocks. Although the "Human Sentiment" has lower average MAE, it improved only 3 stocks and made worse 15 stocks against "Price Only" model. Therefore, integration of the sentiments seems to be not helpful to predict the exact price in support vector regression.

The results of directional accuracy are shown in Table 5.9. In addition to the result of each stock, we also calculated the average of 18 stocks for each model for easy comparison. The results show that the average directional accuracies between these three methods is not so much different.

The problems with regression models is that they try to optimize the error which is the difference between the true values and predicted value. On the other hand, the classification models try to optimize the movement error. Therefore, as we can see later in the results of classification models, the accuracy of classification models will be higher than the directional accuracy of regression models. For example, the accuracy of Price Only method is 52.34% in SVM (as will be shown in Table 5.10 in the next subsection) compared to 47.8% in linear regression and 50.65% in SVR model. In other words, it is not suitable to use regression to predict the stock price movement.

In addition, in practical use, the direction accuracy is more important than MAE. For example, let us suppose that an investor I have 1000 volumes of a stock S. We want to compare two stock prediction models A and B. The stock prediction model A predicts the

Table 5.9: Results of Directional Accuracies of 18 Stocks Using SVR

Stocks	Price Only	Human Sentiment	Sentiment Classification
AAPL	0.3846	0.3846	0.3846
AMZN	0.3659	0.4146	0.3659
BA	0.6341	0.4878	0.5122
BAC	0.3902	0.4878	0.4878
CSCO	0.4634	0.3902	0.4634
DELL	0.4146	0.4634	0.3659
EBAY	0.5854	0.439	0.561
ETFC	0.439	0.4146	0.4146
GOOG	0.439	0.439	0.439
IBM	0.6341	0.6585	0.6098
INTC	0.4878	0.5122	0.5122
KO	0.5366	0.5122	0.561
MSFT	0.5366	0.5366	0.5366
NVDA	0.5854	0.561	0.5854
ORCL	0.6098	0.6098	0.6585
T	0.5854	0.5854	0.4878
XOM	0.5366	0.439	0.439
YHOO	0.4878	0.5122	0.4878
AVERAGE	**0.5065**	0.4915	0.4929

tomorrow price of stock S with the difference compared to the previous price as $-0.1\$$, while the model B predicts with $+0.5\$$. Let us suppose the true difference is $+0.1\$$. The error in the model A is only 0.2, whereas 0.4 in the model B. However, with the model A, the investor I think that the stock price will go down tomorrow, so he or she decides to sell the stock. On the other hand, with the model B, the investor will keep that stock. With the first selection, the investor I losses $0.1 * 1000 = 100\$$ profit, whereas with the second selection, he or she gains $100\$$ profit. Even though he thought he would gain $0.5 * 1000 = 500\$$ with the model B, he or she can only get a smaller profit than the expectation. However, gaining a small profit is better than losing a profit. Therefore, prediction accuracy is important. The next subsection will focus on the prediction of the stock movement and show the performance of the proposed methods.

5.4.3 Evaluation of Stock Movement Prediction

Experiment Setup

We assigned each transaction date a label (up, down) by comparing the price at the current and previous transaction dates. The performance is evaluated by the accuracy metric. Accuracy is the proportion of true results in the test set as shown in Equation (5.5).

$$Accuracy = \frac{tp + tn}{tp + fp + fn + tn} \tag{5.5}$$

where

tp: the number of samples correctly categorized for positive samples.

tn: the number of samples correctly rejected for the negative samples.

fp: the number of samples incorrectly categorized for the positive samples.

fn: the number of samples incorrectly rejected for the negative samples.

It is exactly same as the directional accuracy in Equation (5.4).

For the hyperparameters of LDA, JST and TSLDA, we simply selected symmetric Dirichlet prior vectors, that is all possible distributions are likely equal. We used the default values of these hyperparameters for LDA and JST. Concretely speaking, $\alpha = 0.5$, $\beta = 0.01$ in LDA and $\alpha = \frac{50}{\#topics}$, $\beta = 0.01$, $\gamma = 0.3$ were used in JST. For TSLDA, we set $\alpha = 0.1$, $\lambda = 0.1$, $\beta = 0.01$ and $\gamma = 0.01$.

Evaluation on 18 Stocks

We divided the time series into training set and test set as described in Subsection 5.4.2. In this experiment, we specify the number of the topics K as 50 topics for all the stocks. This assumption would not be appropriate in general. The number of discussed topics may depend on the content of the message board for the individual stocks. Therefore, the appropriate number of hidden topics may be varied for different stocks. However, there is no way to determine the number of topics in the model of LDA, JST and TSLDA. One of the solutions is a grid search trying different number of topics and finding the best value. However, since the running time of the Gibbs Sampling depends on the size of the dataset, it takes very long time to run it repeatedly on a big dataset of 18 stocks for a long period. Therefore, we only conducted a grid search for 5 stocks as explained in the next evaluation. In addition, because of time costing of TSLDA model in a big size dataset, we do not evaluate TSLDA in these 18 stocks. It will be evaluated in the 5 stocks in the next evaluation.

The results of accuracy measure are shown in Table 5.10. In addition to the result of each stock, we also calculated the average of 18 stocks for each model for easy comparison. Using Aspect-based Sentiment feature achieved the best result with 54.41% average accuracy for 18 stocks. As discussed in Section 2.4, degrees of the accuracy of 56% hit rate are often reported as satisfying results for stock prediction. In addition, the number of instances (transaction dates) in the test set of most of other researches is very small, and the number of stock is usually only one stock. In contrast, the advantage of this work is that the training and test data are on a long period (one year) containing many instances, and for many stocks (18 stocks). For some stocks, the accuracies are quite high, such as 71.05% for AMZN stock, 64.47% for DELL stock and so on.

To assess the effectiveness of integrating mood information, we compare our Aspect-based Sentiment method with the Price Only method. The results show that the model using mood information outperforms 2.07% on the average accuracy than the model without mood. Furthermore, comparing the Human Sentiment with Price Only method, it indicated that the prediction accuracy was improved 1.91% by using the sentiments an-

Table 5.10: Results of Accuracies of 18 Stocks Using SVM

Stocks	Baseline Models				Our Models	
	Price Only	**Human Sentiment**	**Sentiment Classification**	**LDA-based Method**	**JST-based Method**	**Aspect-based Sentiment**
AAPL	0.3951	0.5679	0.4938	0.5802	0.5802	0.5432
AMZN	0.4605	0.4868	0.4605	0.5132	0.5921	0.7105
BA	0.6316	0.6053	0.5132	0.5526	0.6316	0.5921
BAC	0.5658	0.5921	0.5658	0.5526	0.5658	0.4474
CSCO	0.5526	0.4474	0.5263	0.4737	0.5132	0.4605
DELL	0.5395	0.5921	0.4737	0.5132	0.4342	0.6447
EBAY	0.5921	0.4605	0.4605	0.5658	0.4079	0.5789
ETFC	0.5789	0.5921	0.5789	0.4868	0.4342	0.5526
GOOG	0.5	0.5658	0.5789	0.5658	0.5395	0.5263
IBM	0.4868	0.4737	0.4868	0.5395	0.4474	0.5526
INTC	0.4474	0.4605	0.4342	0.5	0.4868	0.5263
KO	0.4079	0.4868	0.5132	0.5658	0.5132	0.4474
MSFT	0.5789	0.6579	0.5921	0.5526	0.5526	0.5263
NVDA	0.6053	0.5789	0.6184	0.3947	0.5	0.5395
ORCL	0.4868	0.5263	0.5263	0.5921	0.5	0.5395
T	0.5526	0.4737	0.4868	0.5	0.5658	0.5132
XOM	0.4868	0.6447	0.4868	0.4342	0.5658	0.5395
YHOO	0.5526	0.5526	0.5395	0.5263	0.4474	0.5526
AVERAGE	0.5234	0.5425	0.5187	0.5227	0.5154	**0.5441**

notated by human.

To assess the effectiveness of automatic sentiment analysis and human sentiment, we compare our Aspect-based Sentiment method with the Human Sentiment method. The results show that our automatically extracted sentiment is slightly higher than using the sentiment annotated by human. Therefore, our method is comparable to the human sentiment method. Notice that our method does not use the human annotated sentiment. Hence, it can be applicable for other social media without human annotated sentiment such as Twitter.

The Aspect-based Sentiment method outperformed 2.54%, 2.14% and 2.87% on average accuracy compared to Sentiment Classification, LDA-based Method and JST-based Method, respectively. The LDA-based method and JST-based method seem to be not successful in this experiment. The limitation of these methods is that we have to specify the number of hidden topics in LDA and the number of hidden topics and sentiments in JST.

Although the sentiment information is effective for the stock prediction on average, in the comparison on the individual stocks, the model with sentiment analysis is worse than the price only model for several stocks. There are many possible reasons for it. One reason is that the sentiment might not be a factor which causes the stock price moving. Another reason is that even though sentiment might be one of the factors which affect price moving, the extracted sentiments from the Message Boards do not reflect the price because of the messy, fault comment or fault prediction of human in the posted messages.

A simple assumption about the effectiveness of the sentiment feature is that the sentiment analysis may not provide any additional information if the stock movement can

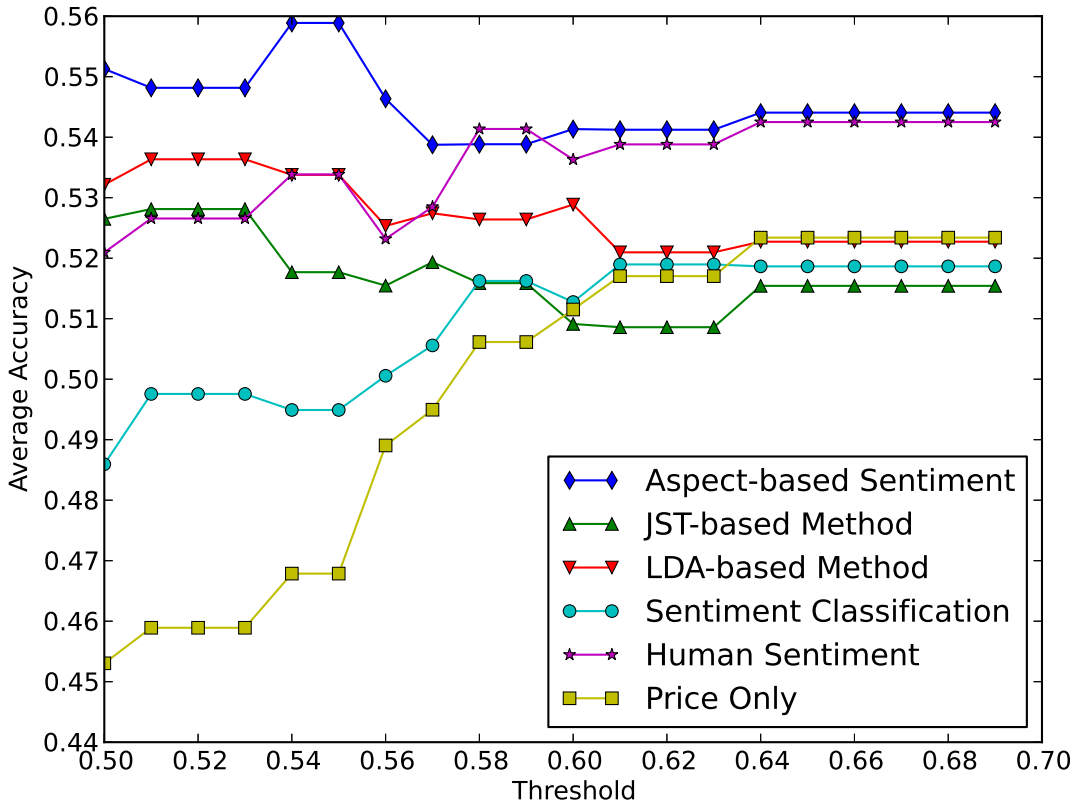

Figure 5.6: Comparison of the Models for Different Threshold α

be predicted well by the historical price only. If the accuracy of the price only model is high, there are trends and historical repetition in the stock. In such cases, only historical prices might be enough to predict, and integration of the sentiment may not improve the accuracy much. On the other hand, if the accuracy of the price only model is low, the stock seems to have no pattern in its history. For such stocks, the use of sentiment may be effective for the prediction.

To investigate the above assumption, we compare the models from another point of view. First, we define a threshold α. If the accuracy of the stock in the Price Only method ($A_{PriceOnly}$) is higher than α, this stock is discarded from the evaluation. In other words, we compared the average accuracy for the stocks where $A_{PriceOnly} < \alpha$. Figure 5.6 shows the average accuracies against various thresholds. It is found that the difference between the models with and without sentiment information becomes greater when α is set smaller. At the threshold 50%, using our Aspect-based Sentiment model improved the accuracy over 9.83% compared to Price Only, over 3.03% compared to Human Sentiment method. In addition, in the most of the thresholds, our method achieved the best accuracy compared with other methods.

Table 5.11: Accuracies of Stock Movement Prediction

Stocks	Price Only	LDA	JST	TSLDA
XOM	0.5000	0.4464	0.5179	**0.5357**
DELL	**0.5893**	0.5357	0.5000	0.5536
EBAY	0.6071	0.6071	0.5000	**0.6429**
IBM	0.4107	0.3929	0.5357	**0.5536**
KO	0.4107	0.5179	0.4643	**0.5357**
AVERAGE	0.5036	0.5000	0.5036	**0.5643**

Evaluation on 5 Stocks

Because of time costing of training TSLDA model in a large amount of the dataset, we only evaluate TSLDA with 5 stocks for which there are the lowest number of messages among 18 stocks. In addition, rather than choosing 50 as the number of topics for all stocks, we investigate various number of topics and chose the best number for each stock based on the development set.

We divided the dataset into three parts: training set from July 23, 2012 to March 31, 2013, development set from April 01, 2013 to April 30, 2013, and test set from May 01, 2013 to July 19, 2013. The label of 'up' and 'down' is assigned to each transaction date by comparing the price of the current and previous dates.

To optimize the number of topics K for each stock, we run the models with four values of K: 10, 20, 50 and 100. The best K is chosen for each stock on the development set, and the systems with the chosen K is evaluated on the test data. The performance of the prediction is measured by the accuracy.

The result of each stock is shown in Table 5.11. In addition, the average of 5 stocks for each model is revealed in the last row of this table for easy comparison. Our model TSLDA-based method outperformed the other methods on the average of the stocks. Table 5.12 shows the number of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) of models for the stocks. For easy comparison, the summation for these five stocks are calculated in the last row.

To assess the effectiveness of integrating mood information, we compare our TSLDA-based method with Price Only method. The results showed that the model using mood information outperformed the model without mood by 3.57%, 3.58%, 14.29% and 12.5% accuracy for XOM, EBAY, IBM and KO stock, respectively. On the other hand, the performance on DELL stock was not improved. It means that the use of the mood does not always make the performance better. The mood from social media could lead to a wrong prediction because of wrong prediction of message writers, fault information and so on. However, TSLDA was better than Price Only method on average of these stocks. In addition, TSLDA can reduce the number of FN, especially for IBM, although FP was not changed in the sum of 5 stocks. Thus, we can conclude that integrating the mood information from social media can help to predict stock price movement more precisely.

Next, let us compare the models for inferring latent topics only (LDA) and topics and

Table 5.12: TP, TN, FP, FN of Stock Movement Prediction

Stocks	**Metrics**	**Price Only**	**LDA**	**JST**	**TSLDA**
XOM	TP	14	13	15	18
	TN	14	12	14	12
	FP	8	10	8	10
	FN	20	21	19	16
DELL	TP	17	13	5	13
	TN	16	17	23	18
	FP	17	16	10	15
	FN	6	10	18	10
EBAY	TP	17	18	20	20
	TN	17	16	8	16
	FP	9	10	18	10
	FN	13	12	10	10
IBM	TP	15	15	7	31
	TN	8	7	23	0
	FP	17	18	2	25
	FN	16	16	24	0
KO	TP	12	14	16	10
	TN	11	15	10	20
	FP	17	13	18	8
	FN	16	14	12	18
Sum	TP	75	73	63	92
	TN	66	67	78	66
	FP	68	67	56	68
	FN	71	73	83	54

Table 5.13: Top Words in Topics of TSLDA

Topic1	**Topic2**	**Topic3**	**Topic4**	**Topic5**	**Topic6**
ko	split	drink	customer	company	country
ceo	stock	coke	budget	competitor	tax
company	share	water	campaign	buy	governor
report	price	produce	promotion	sell	obama
earning	dividend	product	growth	hold	rommey
analyst	year	health	sale	problem	mitt
share	date	juice	volumn	soda	president
news	market	make	come	product	bill
downgrade	time	p.o.s	revenue	people	christian

sentiments (JST and TSLDA) in the stock movement prediction. The accuracy of JST-based method was better than LDA for two stocks (XOM and IBM), worse for three stocks and comparable in the average of five stocks. While, TSLDA-based method outperformed LDA and JST by 2 to 17% in the accuracy for five stocks. TSLDA was also better than LDA and JST on average as shown in Table 5.11. The improvement of the accuracy was derived by increase of TP and decrease of FN. These results indicate that (1) our idea to use both latent topics and sentiments as the features is effective, (2) TSLDA is more appropriate model than JST in stock movement prediction.

Table 5.13 shows examples of highly associated words of some topics for stock KO (Coca-Cola Company) in TSLDA. For example, 'split', 'stock' and 'share' are words highly associated with the hidden topic 2, and 'drink', 'coke' and 'water' are highly associated with the topic 3. The first five hidden topics in Table 7 may represent the management, stock market trading, product, customer care service, competitors of the company, while the last one indicates macroeconomic factors. Table 5.14 shows examples of highly associated words of three sentiments of the hidden topic 1 and 2. For the hidden topic 1, 'growth', 'strong', 'solid' etc. are the words highly associated with the hidden sentiment 3 (which may corresponds to positive class), while 'old', 'tired', 'unreal', etc. with the hidden sentiment 1 (may be negative). We also found that the words with high probabilities in the background distribution were the stop words, punctuations, function words, messy characters written in social media, e.g. '.', 'the', 'and', 'you', '$', 'for' and '?'.

Table 5.15 shows top words in some joint sentiment topic distributions of JST model for stock KO. For example, 'yahoo', 'ko' and 'finance' are highly associated with the joint distribution of hidden sentiment 1 and hidden topic 1. However, it is rather difficult to know which sentiment or topic in this joint distribution actually means.

Table 5.16 shows top words in some topic distributions of LDA model for stock KO. For example, 'nice', 'post' and 'follow' are highly associated words in the hidden topic 1. However, there seems to be no sentiments in these distributions. Only exploiting topics is not enough to know the opinion of people. This could be one of the reasons why our TSLDA model achieved better performance than LDA model.

Table 5.14: Top Words in Sentiments of Topics of TSLDA

Topic1			**Topic2**		
S1	**S2**	**S3**	**S1**	**S2**	**S3**
old	value	grow	down	straight	good
tired	even	strong	tough	warm	long
unreal	difference	solid	troll	informative	more
much	list	gain	breakthrough	interesting	high
obviously	together	full	ex	later	still
much	serve	continue	sugary	responsible	right
not	americans	growth	ep	yeah	sure
helpful	operation	value	richly	used	same
here	get	quarter	major	though	many

Table 5.15: Top Words in Joint Sentiment Topics of JST

S1		**S2**		**S3**	
Topic1	**Topic2**	**Topic1**	**Topic2**	**Topic1**	**Topic2**
yahoo	juice	ko	new	spam	split
ko	minute	buy	american	board	share
finance	maid	get	country	post	date
chart	orange	sell	obama	ignore	stock
free	apple	go	top	idiot	record
fire	drink	make	fall	get	price
website	fruit	money	health	read	august
aone	edit	much	government	another	receive
download	punch	next	place	report	get

Table 5.16: Top Words in Topics of LDA

Topic1	**Topic2**	**Topic3**	**Topic4**	**Topic5**	**Topic6**
nice	ko	coke	buy	good	split
post	time	drink	stock	day	share
follow	year	juice	dividend	back	stock
report	soda	minute	price	work	date
article	sodastream	market	great	week	record
read	point	product	money	give	trade
spam	earning	real	long	move	price
virus	lower	sell	hold	free	august
love	taste	cola	high	call	receive

5.5 Summary

As discussed in Section 2.4, stock price prediction is a very challenging task because the stock prices are affected by many factors. The Efficient Market Hypothesis and random walk theory said that it cannot be predictable with more than about 50% accuracy. On the other hand, some researches specify that the stock market prices can be predicted at some degree. Degrees of accuracy around 56% are often reported as satisfying results.

With the assumption that the integration of the sentiments from the social media can help to improve the predictive ability of models, we have evaluated and compared three feature sets for stock price prediction and seven feature sets for stock movement prediction. In addition, to address the question how automatic sentiment analysis contributes to the prediction, we evaluated the automatic sentiment analysis against the human annotated sentiment. We proposed a new feature topic-sentiment for stock prediction, and extracted it by using three methods (JST-based Method, TSLDA-based Method and Aspect-based Sentiment Method). In addition, the advantage of our experiment was that we conducted our methods for many stocks (18 stocks) and for a long time period of the test set (four months).

In the stock price prediction evaluation, the results show that the integration of sentiments from social media by "Human Sentiment" and "Sentiment Classification" methods seems to be ineffective. However, these integration is helpful for improvement of the directional accuracies in some cases in linear regression method.

In the stock movement prediction evaluation of 18 stocks, the results show that the aspect-based sentiment model outperformed the others in the accuracy measure. Besides, our method is comparable to the method using manually annotated sentiments. Therefore, the automatic sentiment analysis can be the alternative of the manual annotation. When considering only the stocks whose movement are difficult to predict only with the price history, it is revealed that integrating sentiments for the stocks can help to improve the prediction 9.83% accuracy compared to Price Only method, and 3.03% accuracy compared to Human Sentiment method.

On the other hand, in the stock movement prediction evaluation of 5 stocks, the results of the experiments show the effectiveness of our proposed TSLDA-based method. It outperformed the other topic modeling and "Price Only" methods.

The weakness of the LDA-based Method, JST-based Method and TSLDA-based Method model is that we have to specify the number of topics and sentiment beforehand. To overcome of this weakness, a non-parametric topic model which extracts simultaneously topic and sentiment is very useful for stock prediction. This will be done in our future work.

Chapter 6

Conclusion

6.1 Summary of the Dissertation

In this dissertation, we have presented a study on sentiment analysis and its application to stock market prediction. We have proposed new methods for aspect-based sentiment analysis and a framework for the stock market prediction with the sentiments in social media. In addition, we proposed a new topic model, TSLDA, which can extract simultaneously topics and sentiments in the documents. This model was used in the aspect-based sentiment analysis as well as in the sentiment analysis for stock movement prediction.

6.1.1 Aspect-based Sentiment Analysis

We focus on two subtasks of aspect-based sentiment analysis: aspect term polarity identification and aspect category polarity identification. The aspect term polarity identification is a task to identify the sentiment category for a given aspect term in the sentence. In contrast, aspect category polarity identification is a task to classify the sentiment for a given aspect category.

Five methods were employed in the first subtask. First, a simple unsupervised model "ASA w/o RE" was used and further improved toward another model "ASA with RE" by integrating aspect-opinion relations. To extract these relations, we proposed a new tree kernel based on the constituent and dependency trees. The results show that "ASA with RE" outperformed the "ASA w/o RE" by 5.8% accuracy and 4.6% F-measure. Therefore, the integration of aspect-opinion relation extraction is useful for aspect-based sentiment analysis. Furthermore, a new supervised method PhraseRNN was proposed based on the recursive neural network. Our PhraseRNN outperformed "ASA w/o RE", "ASA with RE", RNN and AdaRNN methods for this subtask.

For the second subtask, two topic models which can extract simultaneously topics and sentiments, JST and our proposed TSLDA models, were used to identify the aspect categories and their sentiments. Moreover, a new method was introduced to map the inferred topics/sentiment to human topics/sentiments. The results indicate that our TSLDA achieved better performance JST by 4 to 15% accuracy and 4 to 9% in F-measure

in three review datasets. Thus, our TSLDA is more suitable than JST model for aspect category polarity identification.

6.1.2 Sentiment Analysis for Stock Market Prediction

With the assumption that the sentiments from the social media can help to improve the predictive ability of the models, we have evaluated and compared three feature sets for stock price prediction and seven feature sets for stock movement prediction. In addition, to address the question how automatic sentiment analysis contributes to the prediction, we evaluated the automatic sentiment analysis against the human annotated sentiment. We proposed a new feature topic-sentiment for stock prediction, and extracted it by using three methods (JST-based Method, TSLDA-based Method and Aspect-based Sentiment Method). The results show that the TSLDA-based and aspect-based sentiment methods outperformed others in the accuracy measure. Besides, our method was comparable to the method using manually annotated sentiments. Therefore, the automatic sentiment analysis can be the alternative of the manual annotation. In addition, the important contribution of our experiment was that we evaluated our methods for many stocks (18 stocks) and for a long time period of the test set (four months).

By considering only the stocks whose movement was difficult to predict with only the price history, it was revealed that the integration of the sentiments can help to improve the prediction over 9.83% accuracy compared to Price Only method, and over 3.03% accuracy compared to Human Sentiment method.

6.2 Future Work

In future work, we plan to pursue the following directions:

1. **Improving the aspect-opinion relation extraction by integrating the semantic tree**
   Currently, our tree kernel based model for aspect-opinion relation extraction only used the information from two syntactic trees. Semantic information from semantic trees could improve the extraction of these relations. In future research, a tree kernel based on semantic trees will be considered for aspect-opinion relation extraction. In addition, combination of the syntactic tree and semantic tree for calculating tree kernel will be explored.

2. **Learning the weight parameters for "ASA with RE"**
   In aspect term polarity identification, our "ASA with RE" is an unsupervised model. Instead of using the predefined weights ($weight(a, ow) = 2$ if there exists the relation between the aspect and opinion, and 0 otherwise), we learn the weights from the training dataset. The learnt weights could capture more accurately how the aspect-opinion relations contribute to the sentiment of the aspect. We can expect that the accuracy of the system could be improved with the learnt weights.

3. **Studying the non-parametric topic model for TSLDA**
   The weakness of the LDA-based Method, JST-based Method and TSLDA-based Method is that we have to specify the number of topics and sentiment beforehand. However, there is no appropriate way to specify these number of the topics for each stock. To overcome this weakness, a non-parametric topic model that can infer the number of topics and sentiments automatically is useful to extract the topic and sentiment simultaneously for the stock prediction.

4. **Studying a fine-grained stock prediction model**
   The stock prediction of up or down in each day is useful. However, it may be insufficient for practical use. To make a better strategy decision, the investors and managers of the companies need to know the degree of stock movement. Our model can be extended to predict the degree of the change by setting more fine grained classes such as 'great up', 'little up', 'little down', 'great down' and so on.

5. **Developing a more accurate stock prediction model**
   Instead of using only the historical prices and sentiment derived from social media, we will try to find and integrate more factors which can affect the stock prices. For example, the financial conditions of the company, which can be guessed from the income statement, balance sheet, cash flow, macroeconomic indicators and co-variance between stocks are important factors to be considered in the stock prediction model.

Bibliography

[1] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining Text Data*, pp. 415–463, Springer, 2012.

[2] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and trends in information retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008.

[3] G. Preethi and B. Santhi, "Stock market forecasting techniques: A survey.," *Journal of Theoretical & Applied Information Technology*, vol. 46, no. 1, 2012.

[4] W. Antweiler and M. Z. Frank, "Is all that talk just noise? the information content of internet stock message boards," *The Journal of Finance*, vol. 59, no. 3, pp. 1259–1294, 2004.

[5] R. Tumarkin and R. F. Whitelaw, "News or noise? internet postings and stock prices," *Financial Analysts Journal*, vol. 57, no. 3, pp. 41–51, 2001.

[6] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of Computational Science*, vol. 2, no. 1, pp. 1–8, 2011.

[7] T. T. Vu, S. Chang, Q. T. Ha, and N. Collier, "An experiment in integrating sentiment features for tech stock prediction in twitter," in *24th International Conference on Computational Linguistics*, pp. 23–38, 2012.

[8] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[9] F. Sebastiani, "Machine learning in automated text categorization," *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[10] Y. K. Lee and H. T. Ng, "An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 41–48, Association for Computational Linguistics, 2002.

[11] J. Nivre, J. Hall, J. Nilsson, G. Eryiit, and S. Marinov, "Labeled pseudo-projective dependency parsing with support vector machines," in *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pp. 221–225, Association for Computational Linguistics, 2006.

[12] L.-M. Nguyen, A. Shimazu, and X.-H. Phan, "Semantic parsing with structured svm ensemble classification models," in *Proceedings of the COLING/ACL on Main conference poster sessions*, pp. 619–626, Association for Computational Linguistics, 2006.

[13] T. Mullen and N. Collier, "Sentiment analysis using support vector machines with diverse information sources.," in *EMNLP*, vol. 4, pp. 412–418, 2004.

[14] T. H. Nguyen and K. Shirai, "Aspect-based sentiment analysis using tree kernel based relation extraction," in *Computational Linguistics and Intelligent Text Processing*, pp. 114–125, Springer, 2015.

[15] M. Zhang and H. Li, "Tree kernel-based svm with structured syntactic knowledge for btg-based phrase reordering," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pp. 698–707, Association for Computational Linguistics, 2009.

[16] G. Boella and L. Di Caro, "Extracting definitions and hypernym relations relying on syntactic dependencies and support vector machines.," in *ACL (2)*, pp. 532–537, 2013.

[17] C. Goller and A. Kuchler, "Learning task-dependent distributed representations by backpropagation through structure," in *Neural Networks, 1996., IEEE International Conference on*, vol. 1, pp. 347–352, IEEE, 1996.

[18] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 129–136, 2011.

[19] G. Ifrim, G. Bakir, and G. Weikum, "Fast logistic regression for text categorization with variable-length n-grams," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 354–362, ACM, 2008.

[20] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: a methodology review," *Journal of biomedical informatics*, vol. 35, no. 5, pp. 352–359, 2002.

[21] C. M. Bishop *et al.*, *Pattern recognition and machine learning*, vol. 4. Springer New York, 2006.

[22] T. Haistie, R. Tibshirani, and J. Friedman, "The elements of statistical learning-data mining," *Inference and Prediction: Springer Series in Statistics*, 2009.

[23] A. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Advances in neural information processing systems*, vol. 14, p. 841, 2002.

[24] G. A. Seber and A. J. Lee, *Linear regression analysis*, vol. 936. John Wiley & Sons, 2012.

[25] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, V. Vapnik, *et al.*, "Support vector regression machines," *Advances in neural information processing systems*, vol. 9, pp. 155–161, 1997.

[26] B. Pang, L. Lee, and S. 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*, pp. 79–86, Association for Computational Linguistics, 2002.

[27] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining.," in *LREC*, vol. 10, pp. 1320–1326, 2010.

[28] R. Moraes, J. F. Valiati, and W. P. G. Neto, "Document-level sentiment classification: An empirical comparison between svm and ann," *Expert Systems with Applications*, vol. 40, no. 2, pp. 621–633, 2013.

[29] T. Liu, M. Li, S. Zhou, and X. Du, "Sentiment classification via l2-norm deep belief network," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 2489–2492, ACM, 2011.

[30] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proceedings of the 12th international conference on World Wide Web*, pp. 519–528, ACM, 2003.

[31] B. Ohana and B. Tierney, "Sentiment classification of reviews using sentiwordnet," in *9th. IT & T Conference*, p. 13, 2009.

[32] G. Paltoglou and M. Thelwall, "A study of information retrieval weighting schemes for sentiment analysis," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 1386–1395, Association for Computational Linguistics, 2010.

[33] H. Yu and V. Hatzivassiloglou, "Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences," in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pp. 129–136, Association for Computational Linguistics, 2003.

[34] T. Wilson, J. Wiebe, and R. Hwa, "Just how mad are you? finding strong and weak opinion clauses," in *aaai*, vol. 4, pp. 761–769, 2004.

[35] A. Esuli and F. Sebastiani, "Enhancing opinion extraction by automatically annotated lexical resources," in *Human Language Technology. Challenges for Computer Science and Linguistics*, pp. 500–511, Springer, 2011.

[36] F. Bravo-Marquez, M. Mendoza, and B. Poblete, "Combining strengths, emotions and polarities for boosting twitter sentiment analysis," in *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining*, p. 2, ACM, 2013.

[37] P. J. Stone, D. C. Dunphy, and M. S. Smith, "The general inquirer: A computer approach to content analysis.," 1966.

[38] V. Hatzivassiloglou and K. R. McKeown, "Predicting the semantic orientation of adjectives," in *Proceedings of the 35th annual meeting of the association for computational linguistics and eighth conference of the european chapter of the association for computational linguistics*, pp. 174–181, Association for Computational Linguistics, 1997.

[39] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pp. 347–354, Association for Computational Linguistics, 2005.

[40] M. M. Bradley and P. J. Lang, "Affective norms for english words (anew): Instruction manual and affective ratings," tech. rep., Technical Report C-1, The Center for Research in Psychophysiology, University of Florida, 1999.

[41] F. Å. Nielsen, "A new ANEW: evaluation of a word list for sentiment analysis in microblogs," *CoRR*, vol. abs/1103.2903, 2011.

[42] M. Thelwall, K. Buckley, and G. Paltoglou, "Sentiment strength detection for the social web," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 1, pp. 163–173, 2012.

[43] S. M. Mohammad and P. D. Turney, "Crowdsourcing a word–emotion association lexicon," *Computational Intelligence*, vol. 29, no. 3, pp. 436–465, 2013.

[44] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[45] S. Baccianella, A. Esuli, and F. Sebastiani, "Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining.," in *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*, vol. 10, pp. 2200–2204, 2010.

[46] S. Mohammad, C. Dunne, and B. Dorr, "Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pp. 599–608, Association for Computational Linguistics, 2009.

[47] S.-M. Kim and E. Hovy, "Determining the sentiment of opinions," in *Proceedings of the 20th international conference on Computational Linguistics*, p. 1367, Association for Computational Linguistics, 2004.

[48] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 168–177, ACM, 2004.

[49] N. Kaji and M. Kitsuregawa, "Building lexicon for sentiment analysis from massive collection of html documents.," in *EMNLP-CoNLL*, pp. 1075–1083, 2007.

[50] N. Jakob and I. Gurevych, "Extracting opinion targets in a single-and cross-domain setting with conditional random fields," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1035–1045, Association for Computational Linguistics, 2010.

[51] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[52] W. Jin, H. H. Ho, and R. K. Srihari, "A novel lexicalized hmm-based learning framework for web opinion mining," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 465–472, Citeseer, 2009.

[53] W. Jin, H. H. Ho, and R. K. Srihari, "Opinionminer: a novel machine learning system for web opinion mining and extraction," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1195–1204, ACM, 2009.

[54] D. Freitag and A. McCallum, "Information extraction with hmm structures learned by stochastic optimization," *AAAI/IAAI*, vol. 2000, pp. 584–589, 2000.

[55] Q. Su, X. Xu, H. Guo, Z. Guo, X. Wu, X. Zhang, B. Swen, and Z. Su, "Hidden sentiment association in chinese web opinion mining," in *Proceedings of the 17th international conference on World Wide Web*, pp. 959–968, ACM, 2008.

[56] A.-M. Popescu and O. Etzioni, "Extracting product features and opinions from reviews," in *Natural language processing and text mining*, pp. 9–28, Springer, 2007.

[57] G. Qiu, B. Liu, J. Bu, and C. Chen, "Expanding domain sentiment lexicon through double propagation.," in *IJCAI*, vol. 9, pp. 1199–1204, 2009.

[58] G. Qiu, B. Liu, J. Bu, and C. Chen, "Opinion word expansion and target extraction through double propagation," *Computational linguistics*, vol. 37, no. 1, pp. 9–27, 2011.

[59] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631, p. 1642, Citeseer, 2013.

[60] L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, and K. Xu, "Adaptive recursive neural network for target-dependent twitter sentiment classification," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 49–54, 2014.

[61] L. Dong, F. Wei, M. Zhou, and K. Xu, "Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis," in *Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI). AAAI*, 2014.

[62] S.-M. Kim and E. Hovy, "Extracting opinions, opinion holders, and topics expressed in online news media text," in *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, pp. 1–8, Association for Computational Linguistics, 2006.

[63] C. J. Fillmore, C. R. Johnson, and M. R. Petruck, "Background to framenet," *International journal of lexicography*, vol. 16, no. 3, pp. 235–250, 2003.

[64] S. Brody and N. 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*, pp. 804–812, Association for Computational Linguistics, 2010.

[65] Y. Jo and A. H. Oh, "Aspect and sentiment unification model for online review analysis," in *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 815–824, ACM, 2011.

[66] C. Lin and Y. He, "Joint sentiment/topic model for sentiment analysis," in *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 375–384, ACM, 2009.

[67] W. X. Zhao, J. Jiang, H. Yan, and X. Li, "Jointly modeling aspects and opinions with a maxent-lda hybrid," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 56–65, Association for Computational Linguistics, 2010.

[68] H. Lakkaraju, C. Bhattacharyya, I. Bhattacharya, and S. Merugu, "Exploiting coherence for the simultaneous discovery of latent facets and associated sentiments," in *Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, April 28-30, 2011, Mesa, Arizona, USA*, pp. 498–509, SIAM / Omnipress, 2011.

[69] E. F. Fama, L. Fisher, M. C. Jensen, and R. Roll, "The adjustment of stock prices to new information," *International economic review*, vol. 10, no. 1, pp. 1–21, 1969.

[70] E. F. Fama, "Efficient capital markets: Ii," *The journal of finance*, vol. 46, no. 5, pp. 1575–1617, 1991.

[71] S. Walczak, "An empirical analysis of data requirements for financial forecasting with neural networks," *Journal of management information systems*, vol. 17, no. 4, pp. 203–222, 2001.

[72] B. Qian and K. Rasheed, "Stock market prediction with multiple classifiers," *Applied Intelligence*, vol. 26, no. 1, pp. 25–33, 2007.

[73] R. P. Schumaker and H. Chen, "Textual analysis of stock market prediction using breaking financial news: The azfin text system," *ACM Trans. Inf. Syst.*, vol. 27, pp. 12:1–12:19, Mar. 2009.

[74] J. Si, A. Mukherjee, B. Liu, Q. Li, H. Li, and X. Deng, "Exploiting topic based twitter sentiment for stock prediction," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, pp. 24–29, The Association for Computer Linguistics, 2013.

[75] G. Tsibouris and M. Zeidenberg, "Testing the efficient markets hypothesis with gradient descent algorithms," in *Neural Networks in the Capital Markets*, pp. 127–136, Wiley: Chichester, 1995.

[76] Y. Zuo and E. Kita, "Stock price forecast using bayesian network," *Expert Systems with Applications: An International Journal*, vol. 39, no. 8, pp. 6729–6737, 2012.

[77] Y. Zuo and E. Kita, "Up/down analysis of stock index by using bayesian network," *Engineering Management Research*, vol. 1, no. 2, pp. 46–52, 2012.

[78] R. P. Schumaker and H. Chen, "A quantitative stock prediction system based on financial news," *Information Processing & Management*, vol. 45, no. 5, pp. 571–583, 2009.

[79] R. Luss and A. d'Aspremont, "Predicting abnormal returns from news using text classification," *Quantitative Finance*, no. ahead-of-print, pp. 1–14, 2012.

[80] S. Samangooei, D. Preotiuc, J. Li, M. Niranjan, N. Gibbins, and T. Cohn, "D3. 1.1 regression models of trends in streaming data," tech. rep., Technical report, University of Sheffield, 2012.

[81] X. Zhang, H. Fuehres, and P. A. Gloor, "Predicting stock market indicators through twitter "I hope it is not as bad as I fear"," *Procedia-Social and Behavioral Sciences*, vol. 26, no. 0, pp. 55–62, 2011.

[82] J. Bollen, A. Pepe, and H. Mao, "Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena," in *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pp. 450–453, 2011.

[83] B. Xie, R. J. Passonneau, L. Wu, and G. Creamer, "Semantic frames to predict stock price movement.," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pp. 873–883, 2013.

[84] M. Rechenthin, W. N. Street, and P. Srinivasan, "Stock chatter: Using stock sentiment to predict price direction," *Algorithmic Finance*, vol. 2, no. 3, pp. 169–196, 2013.

[85] N. Balakrishnan, *Continuous multivariate distributions*. Wiley Online Library, 2001.

[86] N. Balakrishnan and V. B. Nevzorov, *A primer on statistical distributions*. John Wiley & Sons, 2004.

[87] J. Huang, "Maximum likelihood estimation of dirichlet distribution parameters," *CMU Technique Report*, 2005.

[88] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[89] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 50–57, ACM, 1999.

[90] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proceedings of the National Academy of Sciences*, vol. 101, no. suppl 1, pp. 5228–5235, 2004.

[91] T. Minka and J. Lafferty, "Expectation-propagation for the generative aspect model," in *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pp. 352–359, Morgan Kaufmann Publishers Inc., 2002.

[92] A. Culotta and J. S. Sorensen, "Dependency tree kernels for relation extraction," in *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)* (D. Scott, W. Daelemans, and M. A. Walker, eds.), pp. 423–429, ACL, 2004.

[93] F. Reichartz, H. Korte, and G. Paass, "Dependency tree kernels for relation extraction from natural language text," in *Machine Learning and Knowledge Discovery in Databases* (W. L. Buntine, M. Grobelnik, D. Mladenic, and J. Shawe-Taylor, eds.), vol. 5782, pp. 270–285, Springer, 2009.

[94] Y. Wu, Q. Zhang, X. Huang, and L. Wu, "Phrase dependency parsing for opinion mining," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1533–1541, ACL, 2009.

[95] D. Zelenko, C. Aone, and A. Richardella, "Kernel methods for relation extraction," *Journal of Machine Learning Research (JMLR)*, vol. 3, pp. 1083–1106, 2003.

[96] L. Sun and X. Han, "A feature-enriched tree kernel for relation extraction," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 61–67, Association for Computational Linguistics, 2014.

[97] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[98] T. T. Nguyen, A. Moschitti, and G. Riccardi, "Convolution kernels on constituent, dependency and sequential structures for relation extraction," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1378–1387, ACL, 2009.

[99] R. C. Bunescu and R. J. Mooney, "A shortest path dependency kernel for relation extraction," in *Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pp. 724–731, ACL, 2005.

[100] N. Kobayashi, K. Inui, and Y. Matsumoto, "Extracting aspect-evaluation and aspect-of relations in opinion mining," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 1065–1074, ACL, 2007.

[101] T. Joachims, *Text categorization with support vector machines: Learning with many relevant features*. Springer, 1998.

[102] T. H. Nguyen and K. Shirai, "Text classification of technical papers based on text segmentation," in *Natural Language Processing and Information Systems - 18th International Conference on Applications of Natural Language to Information Systems, NLDB 2013, Salford, UK, June 19-21, 2013.*, pp. 278–284, 2013.

[103] A. J. Smola and S. Vishwanathan, "Fast kernels for string and tree matching," in *Advances in Neural Information Processing Systems 15 (NIPS)* (S. Becker, S. Thrun, and K. Obermayer, eds.), pp. 585–592, MIT Press, 2003.

[104] M. Collins and N. Duffy, "New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL)*, pp. 263–270, ACL, 2002.

[105] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60, 2014.

[106] M. D. Smucker, J. Allan, and B. Carterette, "A comparison of statistical significance tests for information retrieval evaluation," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (CIKM)*, pp. 623–632, ACM, 2007.

[107] M. Pontiki, H. Papageorgiou, D. Galanis, I. Androutsopoulos, J. Pavlopoulos, and S. Manandhar, "Semeval-2014 task 4: Aspect based sentiment analysis," in *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pp. 27–35, 2014.

[108] J. Wagner, P. Arora, S. Cortes, U. Barman, D. Bogdanova, J. Foster, and L. Tounsi, "Dcu: Aspect-based polarity classification for semeval task 4," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 223–229, 2014.

Publications

Journal Article

[1] **Thien Hai Nguyen**, Kiyoaki Shirai and Julien Velcin. "Sentiment Analysis on Social Media for Stock Movement Prediction". *International Journal of Expert Systems With Applications (ESWA)*, 2015. (Accepted)

Refereed Conference Papers

[2] **Thien Hai Nguyen** and Kiyoaki Shirai. "PhraseRNN: Phrase Recursive Neural Network for Aspect-based Sentiment Analysis". *The Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2509-2514, 2015, September.

[3] **Thien Hai Nguyen** and Kiyoaki Shirai. "Topic Modeling based Sentiment Analysis on Social Media for Stock Market Prediction". *The 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*, pp. 1354-1364, 2015, July.

[4] **Thien Hai Nguyen** and Kiyoaki Shirai. "Aspect-Based Sentiment Analysis Using Tree Kernel Based Relation Extraction". *The 16th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pp. 114-125, 2015, April.

Refereed Conference Papers (not related to the dissertation)

[5] **Thien Hai Nguyen** and Kiyoaki Shirai. "Text Classification of Technical Papers Based on Text Segmentation". *The 18th International Conference on Application of Natural Language to Information Systems (NLDB)*, Vol. 7934, pp. 278-284, 2013, June.

Non-Refereed Conference Papers (not related to the dissertation)

[6] **Thien Hai Nguyen** and Kiyoaki Shirai. "Text Classification of Technical Papers Focusing on Title and Important Segments". *Special Interest Group of Natural Language Processing (SIGNL)*, Vol. 2013-SIGNL-211, No.15, pp. 1-8, 2013, May.

[7] **Thien Hai Nguyen** and Kiyoaki Shirai. "Text Classification of Technical Papers Using Text Segmentation". *The 8th International Conference on Natural Language Processing (JapTAL)*, Poster Volume, 2012, October.