

# FEATURE-BASED SENTIMENT ANALYSIS WITH ONTOLOGIES

Berk Taner

Submitted to the Graduate School of Sabancı University  
in partial fulfillment of the requirements for the degree of  
Master of Science

Sabancı University

July, 2011

FEATURE-BASED SENTIMENT ANALYSIS WITH ONTOLOGIES

Approved by:

Assoc. Prof. Dr. Yücel Saygın  
(Thesis Advisor)



Assist. Prof. Dr. Selim Balcısoy



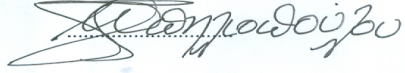
Assoc. Prof. Dr. Mehmet Keskinöz



Assoc. Prof. Dr. Albert Levi



Prof. Dr. Myra Spiliopoulou



Date of Approval: 05/08/2011

© Berk Taner 2011

All Rights Reserved

# ONTOLOJİ KULLANARAK ÖZELLİK TABANLI DUYGU ANALİZİ

Berk Taner

Bilgisayar Bilimi ve Mühendisliği, Yüksek Lisans Tezi, 2011

Thesis Supervisor: Yücel Saygın

**Keywords: Duygu Analizi, Doğal Dil İşleme**

## Özet

Duygu analizi, son yıllarda üzerinde çok fazla araştırma yapılan bir alandır. Duygu analizi temel olarak, büyük veya çok sayıda yazılı metinden olumlu ve olumsuz görüşleri çıkartmaktır. Bu alanda yapılan çalışmalar alt dallara ayrılmıştır. Özellik tabanlı duygu analizi, bir konuda belirli özelliklere yönelik duyguları bulmayı amaçlar. Araştırmaların ilk yıllarında, ticari ürünler ve bu ürünlerin özellikleri üzerine yoğunlaşmıştır. Ürünlerin özellikleri otomatik olarak tespit edilebilmeye başlandığında; bu özellikler için duygu analizi yapılmaya da başlanmıştır. Semantik analiz ve ontoloji konseptlerinin gelişimiyle birlikte, özellik analizi alanında yeni çalışmalar yapılmıştır. Doğal dil işleme teknikleri üzerine yapılan çalışmalar da özellik tabanlı duygu analizi konusuna katkı sağlamaktadır.

Bu tez çalışması özellik tabanlı duygu analizi için doğal dil işleme tekniklerini ve kelimeler arasındaki bağlantıları kullanan bir çerçeve oluşturmak, ontoloji yapısını kullanarak karşılaştırılabilir ve okunabilir sonuçlar oluşturmayı amaçlamaktadır.

# FEATURE-BASED SENTIMENT ANALYSIS WITH ONTOLOGIES

Berk Taner

Computer Science and Engineering, Master's Thesis, 2011

Thesis Supervisor: Yücel Saygın

**Keywords:** Sentiment Analysis, Opinion Mining, Feature-Based Sentiment.

## Abstract

Sentiment analysis is a topic that many researchers work on. In recent years, new research directions under sentiment analysis appeared. Feature-based sentiment analysis is one such topic that deals not only with finding sentiment in a sentence but providing a more detailed analysis on a given domain. In the beginning researchers focused on commercial products and manually generated list of features for a product. Then they tried to generate a feature-based approach to attach sentiments to these features. With the emergence of semantic analysis and ontologies, we now have different domain ontologies created for other purposes that can be used to find features in a domain. Also, Natural Language Processing matured in recent years and allow us to analyze a paragraph in more detail.

This thesis aims to propose a framework for feature-based sentiment analysis that uses NLP techniques to analyze grammatical dependencies between words in a sentence, use ontology representation to model domains, polarity information and results separately, and producing easily readable and comparable summaries as output.

To my mother...

## **Acknowledgements**

First and foremost, I'd like to thank my family, Günergin and Berrin for their unending love and support through my life. Without them, none of this would not be possible.

I also like to thank my supervisors Yücel Saygın and Dilek Tapucu for their support and guidance during my research.

I'll always remember my time on FENS 2014 and all the people that made that lab a wonderful place. It was a pleasure to be part of such a great team.

I also express my gratitude to UBIPOL project for allowing me to tackle an amazing problem and for giving support to my thesis. I also thank my thesis jury for their time and suggestions.

Last, but not least, I'd like to thank Albert Levi for his guidance during all my undergraduate and grad years.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Current State of the Art</b>	<b>3</b>
2.1	Classifying Subjectivity . . . . .	3
2.2	Classifying Sentiment . . . . .	4
2.3	Feature-Based Sentiment Analysis . . . . .	5
2.3.1	Automatic-Feature Extraction . . . . .	5
2.3.2	Feature-Based Opinion Matching . . . . .	6
2.4	Motivation . . . . .	7
<b>3</b>	<b>Problem Definition</b>	<b>9</b>
<b>4</b>	<b>Methodology</b>	<b>11</b>
4.1	Evolution of Our Framework . . . . .	11
4.1.1	GATE . . . . .	11
4.1.2	GATE+ . . . . .	11
4.1.3	Our Framework (Guinevere) . . . . .	12
4.2	Architecture . . . . .	12
4.2.1	Data Structures . . . . .	13
4.2.1.1	Parse Token . . . . .	13
4.2.1.2	Parse Sentence . . . . .	14
4.2.1.3	Parse Phrase . . . . .	14
4.2.2	Modules . . . . .	14
4.2.2.1	NLP Module . . . . .	14
4.2.2.1.1	Part-Of-Speech Tags . . . . .	15
4.2.2.1.2	Parse Tree . . . . .	16
4.2.2.1.3	Dependency Analysis . . . . .	16
4.2.2.2	Ontology Module . . . . .	17
4.3	Polarity Placement . . . . .	20
4.3.1	Dependency Rules . . . . .	21
4.3.1.1	Dependencies That have a Direct Impact . . . . .	21



4.3.1.2	Negators . . . . .	22
4.3.2	Object of Partial Modifiers . . . . .	22
4.3.3	Noun Dependencies . . . . .	24
4.4	Scorecard Representation . . . . .	24
4.5	Finding Overall Sentiment . . . . .	25
<b>5</b>	<b>Experiments</b>	<b>27</b>
5.1	Test Data Set . . . . .	27
5.1.1	Test Data Integrity . . . . .	28
5.2	Performance Metrics . . . . .	30
5.3	Performance Evaluation . . . . .	31
5.4	Performance Discussion . . . . .	32
5.5	Use Cases . . . . .	33
<b>6</b>	<b>Conclusion and Future Works</b>	<b>36</b>

# List of Figures

4.1	Overview of Our Framework . . . . .	13
4.2	Overview of NLP Module . . . . .	15
4.3	Overview of Ontology Module . . . . .	17
4.4	Overview of Our Ontology Structure . . . . .	18
5.1	Distribution of Aspect Ratings for Overall Rating 1. X-axis shows star-ratings and y-axis shows number of comments. . . . .	28
5.2	Distribution of Aspect Ratings for Overall Rating 3. X-axis shows star-ratings and y-axis shows number of comments. . . . .	29
5.3	Distribution of Aspect Ratings for Overall Rating 5. X-axis shows star-ratings and y-axis shows number of comments. . . . .	30

# List of Tables

4.1	Attributes for Parse Token . . . . .	13
4.2	Attributes for Parse Sentence . . . . .	14
4.3	Attributes for Parse Phrase . . . . .	14
4.4	A Sample Scorecard Produced by Our System . . . . .	24
4.5	A Sample Scorecard Log Produced by Our System . . . . .	25
4.6	Table of Ratings . . . . .	26
5.1	Evaluation Results for Aspects . . . . .	31
5.2	Evaluation Results with Different Ratings . . . . .	32
5.3	Detection Accuracy for Features . . . . .	32

# Chapter 1

## Introduction

Since the dawn of time, human beings held different opinions on a subject. As technology progressed from writing to paper, paper to printing press. printing press to radios, radios to Television, and finally Television to World Wide Web; people had an ever increasing opportunity to express their opinions.

With the introduction of World Wide Web (WWW) into our daily lives: the people gained a powerful tool: ability to comment. Over the years this has manifested in different formats: comments on newspaper articles, blogs product reviews and recently twitter & social web.

All of these data available on the web opened way to a new research direction - Sentiment Analysis. Over the years, researchers tended to focus on one aspect of sentiment analysis. This led to the subject to be divided into sub-problems. Liu [14] has categorized current research tracks as follows:

- The Problem of Sentiment Analysis
- Sentiment and Subjectivity Classification
- Feature-Based Sentiment Analysis
- Sentiment Analysis of Comparative Sentences
- Opinion Search and Retrieval
- Opinion Spam and Utility of Opinions

This thesis focuses on Feature-Based Sentiment Analysis and ultimately aims to propose a complete framework dedicated to this problem. The framework has two parts: a natural language processing module handling polarity analysis and an ontology module with our custom polarity words database. The framework also produces a customizable feature-based sentiment analysis report with detailed sentiment information for each feature.

The main contributions of this thesis can be summarized as follows:

- Propose novel techniques and methodologies to determine polarity of a sentence. This part also focuses on placing polarity on the correct word(s) in a sentence in order to enhance the accuracy of feature-based analysis in the later stages.
- Propose a custom ontology for polarity analysis, with custom built polarity words database constructed by SentiWordNet project as well as manual human scoring. This polarity ontology can be integrated and used with other domain-ontologies for tackling different problems.
- Produced and tested a framework with NLP modules, Domain, Polarity and Score-card Ontologies. This framework produces a feature-based sentiment report as a final product. This feature-based report can be customized to focus on different features in a domain.

All these contributions work together to create a framework that can do feature based sentiment analysis on different domains with different ontologies. For the experiments we've used our TripAdvisor ontology, but it can easily be generalized to other domains with different ontologies.

Outline of the rest of the thesis is as follows: In Chapter 2, we give an overview of sentiment analysis research with focus on current state of the art on feature-based sentiment analysis. In Chapter 3 we give a definition of sentiment analysis and it's problems in general. Chapter 4 elaborates on proposed NLP techniques, our analyzing and scoring system and our proposed ontology structure. Chapter 5 deals with the TripAdvisor dataset, case studies and results of proposed framework. Finally on Chapter 6, we summarize the results and implications, and propose some future work on the subject.

## Chapter 2

### Current State of the Art

Prior to World Wide Web, distributing content over the globe was a big problem for small publishers. It was the era of newspapers, periodicals and magazines. With the emergence of WWW, people were empowered with the power of publishing. Now there are many blogs specialized in reviewing certain niches, there are forums for questions, there are social frameworks like twitter for publishing, debating and reviewing products. With this shift in content creation, a huge amount of content started to accumulate over the years. This data-set contained opinions on movies, products, services, and on many other domains. Opinion mining, or sentiment analysis, deals with the problem of extracting sentiment information from given data-sets. In order to solve this problem; ideas and methods from different fields such as Information Retrieval, Natural Language Processing and Machine Learning has been used together.

There are three important sub-topics that needs to be discussed that makes up Feature-Based Sentiment Analysis. Each will be elaborated along with important papers published in their respective fields.

#### 2.1 Classifying Subjectivity

One of the first challenges was to be able to separate content that contains an *opinion*, or subjective content, from objective content. The first step towards subjectivity classification is to find semantic orientation of words. Hatzivassiloglou et al. [8] did an example of such a work focusing on adjectives.

Finn et al. [5] proposed an automatic classifier that can distinguish between documents consisting of opinions and facts. In order to do this they have proposed three different classification techniques. (1) Bag-of-words approach. They generated a list of keywords for the domain of documents. They evaluated the subjectivity by the presence or the absence of a vector of keywords. (2) They made a vector of part-of-speech tags and evaluated a documents subjectivity by the count of certain POS tags. (3) They used

hand-crafted features such as length of sentences and count of certain stop words.

Riloff et al. [19] proposes a subjectivity classifier by training a Naive Bayes classifier by using a list of subjective nouns to determine subjectivity of a document. One other important aspect of this paper is to propose using nouns to determine subjectivity.

At the current state, usage of machine learning algorithms are the standard, and researchers are looking for creating more efficient classifiers using different machine learning approaches such as multi-view learning.[21]. For the past couple of years, there is an increased interest in subjectivity classification in other languages as well.

## 2.2 Classifying Sentiment

After the data-set has been classified into subjective and objective subsets; the sentiment of the document (positive or negative) must be determined. Commonly a set of opinion words are identified and sentiment orientation of a sentence is made determined by evaluation of these words using different techniques. One way is to start with a small set of labeled lexicons and use an unsupervised learning algorithm to grow the sentiment words list as [8] proposed for adjectives. However, with the increased connectivity, open-source projects based on online collaboration exists. One such project is SentiWordNet [1] that provide sentiment orientation for lexicons (words) based on contributions from a high number of users.

As mentioned before the difference lies how the list of opinion words are prepared. One such approach is Dictionary-based approach, where a small set of seed opinion words are selected; and then the list is grown through machine learning algorithms.

Hu et al. [10] proposed one such approach by gathering a seed list of adjectives from SentiWordNet and growing the set using a learning algorithm. They also define *effective opinion*, the closest adjective to a manually labeled feature in a sentence has a slightly bigger weight than other opinion words. They calculate sentiment orientation of a sentence by comparing positively oriented words to negatively oriented words. If there is a tie, effective opinions act as tie breakers. One shortcoming of such approach is that opinion words sometimes domain specific and their semantic orientation may differ. A *big car* may have a positive orientation whereas a *big camera* may have a negative orientation. Corpus-based approach has been developed to have a better domain awareness while determining sentiment orientation of words. This approach aims to grow the list by analyzing patterns in a sentence.

An example has been proposed on [8] where the seed list is grown by analyzing the conjunctions in a sentence. As an example, if the sentence "*This is a big and heavy bag*" is given and *big* is labeled with negative orientation, then the list will add *heavy* as a negative orientation word.

## 2.3 Feature-Based Sentiment Analysis

This sub-topic deals with not only classifying sentiment of a given document, but aims to extract sentiments about certain aspects of a subject. There are two main sub problems in feature-based sentiment analysis and they will be given along with brief summaries of important papers published in the field on next subsections.

### 2.3.1 Automatic-Feature Extraction

Identifying features and feature related keywords are important for feature-based sentiment analysis since it is labor-intensive to manually generate domain-specific feature keywords list. Given the abundance of input data (comments, reviews, etc.) researchers aimed to solve it by leveraging NLP techniques and classifiers.

Hatzivassiloglou et al.[8] proposed to start with a small set of domain related keywords as a seed list. They focused solely on adjectives on they research. They scanned a comment for conjunctions and linkages between adjectives. They first found out that %77 of all conjunctions tie together adjectives of same orientation. The only exception was the linkage word *but*. *But* links two adjectives with different orientations. With these information in mind they analyzed a sentence for any linkages and identified candidate words. Then using a clustering algorithm the classify the word as positive or negative. This paper is important since it is one of the first papers that deal with the problem of expanding feature-keyword sets.

Hu et al. [10] focused on extracting explicitly mentioned features from nouns and noun phrases.They use association mining to find frequently used nouns and noun phrases as candidates. They define frequent phrases as a phrase with a threshold of %1 frequency over entire dataset. One important concept they introduce is *pure-support*. *Pure-support* is determined by the number of sentences a feature candidate appears as noun or a noun phrase, where there are no superset of feature is present. This way they can differentiate between *life* and *battery life*. Their work is important since they propose an efficient algorithm for nouns and noun phrases.

Poppescu et al. [17] developed a feature-based opinion miner called OPINE. It extends Hu's system in order to handle adjectives, adverbs, nouns and verbs. OPINE first identifies feature keywords and then uses a Web-based point-wise mutual information system to decided if the candidate is valid or not. In order to do so, it generates *discriminator phrases* (eg is a scanner) and searches the web for given feature and discriminator. If the results are above a threshold, it is counted as a valid feature. Their claim is that they improved Hu's system precision by %22 while losing %3 recall.

Qui et al. [18] took a different approach. Rather than starting a seed list for features, the used a seed list of sentiment words. Their reasoning is that, in a subjective sentence



if a sentiment word is present, there is a high probability that it is somehow connected to a feature in the sentence. Then they defined sequences of dependencies that identify other sentiment words or features. One of their limitations is that they are constrained to adjective words. Since they haven't used the same dataset with Hu and Poopescu, where is no way of comparing their performances. One other limitation is that, their method is only efficient in medium size corpora. This paper is important since it is one of the first attempts at inferring rules using dependencies in a sentence.

Zhang et al. [24] aims to extract noun feature words from a given corpus. They aimed to improve on Qui by introducing two new patterns in feature detection, namely *part-of* and *no* patterns. First pattern leverages linkages like *of*, *is*, *on* to determine new features part of a class. *no* patterns aims to find phrases like "no (noun phrase)". They use two metrics, feature frequency and feature relevance to assess candidates. In order to find feature relevance they use a web page ranking algorithm to rank candidates. Since they did not use the same dataset as Qui, there is no way to compare them.

This section gives an overview of different approaches to finding features from a given dataset. In next section we will be looking at the research that ties opinions with feature words.

### 2.3.2 Feature-Based Opinion Matching

Assigning polarity to features is the second problem in feature-based sentiment analysis. All of the research done mentioned before contributes to the solution of this problem. We will revisit some of the papers mentioned before-hand with some other important publications that are related to our topic.

Hu et al. [10] reasons that the nearest opinion word in a sentence with the feature word, determines the sentiment on the feature. They proposed some simple rules, like changing the sentiment of the opinion word if *but* is encountered. They assume that generally the nearest opinion word reflects the sentiment of each feature in the sentence. One major drawback of this approach is that they cannot handle complex sentences due to opinion word and sentiment being far from each other and they also propose no way to quantify the polarity for a feature.

Zhuang et al. [26] proposed a set of template rules to generate feature-opinion pairs. Their template focuses on adjectives and noun phrases. They first find candidates for feature and opinion words separately. In this candidate evaluation process, they discard infrequent candidates from their set. Afterwards, they apply their template to generate opinion-feature pairs. As they mentioned in their paper; one major drawback is that people tend to use different words to express their sentiments, and this leads to some valid candidates to be discarded as they were found frequently in the data set. Their experiments yielded average precision of 0.65 and recall of 0.548. This is still not good

enough for fine-grained analysis required on feature-based sentiment analysis.

Zhao et al. [25] were among the first to use ontology on sentiment analysis. They mainly use ontology to define a domain and some features. Then they try to extend knowledge on feature keywords. They also use SentiWordNet and a custom scoring algorithm to translate polarity information obtained into a floating point number. Then they take negation and conjunction words into account to calculate the polarity on a feature. Since they worked on a set of 120 reviews taken from IMDB and did not present precision or recall but simple accuracy, we cannot pass judgement on their accuracy. This paper's main importance lies in taking domain specific information into account while doing feature-based analysis.

Zhang et al. [23] focused on noun features that express opinions. They proposed a scoring algorithm where the effect of an opinion word decreases as the distance between the opinion and the feature word increases. They also defined simple rules for negation, but-clauses and increasing or decreasing rules. The last one is intuitive where they identified polarity words that diminish the sentiment strength of a polarity word. Words like "*decrease*", "*diminish*", "*prevent*" and whenever one of these words are seen, the strength of the sentiment is changed. Surprisingly, their experiments yielded low precision (below 0.5) but high recall (over 0.80 on the average). It is our belief that having a high precision value is critical on feature-based sentiment analysis.

## 2.4 Motivation

There are important concepts mentioned in the previous works that motivated us to bring together and improve on different approaches in a complete framework for feature-based analysis. First we wanted use ontology to effectively represent domain-specific knowledge. Second, we wanted to investigate NLP techniques that will place polarity correctly on features. Third, we wanted to summarize polarity information from SentiWordNet into a simpler format and had human annotators go over it. Fourth, we wanted to propose an easily extendable polarity ontology, to help other research groups. Last, we wanted to propose a new way of summarizing feature-based sentiment results in such a way that it is easily searchable and comparable.

Motivated by these goals we made some design decisions on our framework. We strongly felt that Feature extraction and Polarity placement on features are two related but different topics and left feature-extraction as future work. As mentioned before, there are many published works and tools on this topic, and they can work independently yet easily incorporated to our framework by extending the domain ontology. Instead we focused on producing a polarity lexicon that has separate lists for adjectives, adverbs, nouns and verbs. We also researched dependency relations between words in a sentence, and come up with novel rules that can transfer polarity from an opinion word to a feature word. This

way we aimed to achieve high precision rates and since we have worked with only a seed list of features and feature-keywords, we did not aim for high recall rates. And in the end we proposed a tabular structure we call scorecards to summarize our results.

The following section will define important concepts in our framework before we elaborate on our methodology.

## Chapter 3

### Problem Definition

Sentiment analysis aims to gather the sentiment from a bulk of text. Feature-based sentiment analysis requires a more fine-grained approach; first the features must be identified and the polarity must be placed correctly. Below we give an example comment from Tripadvisor dataset.

”(1) We visited the resort in March 08 - me, my wife and 2 kids. (2) We all had a great holiday. (3) The speciality restaurants are excellent the best we have had in the Dominican R. (4) Beautiful beach. (5) Good selection of pools - all very well maintained. (6) Nice big rooms, well serviced. (7) Great weather. (8) Bad points - lobby bar service sometimes bit slow and could do with more staff - unfortunately if you flashed the dollars there was more chance of getting served quicker. ”

This eight sentence review contains both positively and negatively oriented sentences. What this thesis focuses on is the sentiment orientation of domain features. In this case our domain is a hotel. On feature-based sentiment analysis, the connection between features and sentiment is done by finding and tagging keywords belonging to the feature. Sentence (3) expresses a positive sentiment towards Service feature. Sentence (4) mentions Location feature by mentioning beach whereas sentence (5) mentions pools. Sentence (8) contains sentiment on more than one feature. In order to do an efficient feature-based analysis, we must first define what a feature is and how to store keywords.

**Definition (Ontology) :** As described by Gruber [7]

an *Ontology* is an explicit specification of a conceptualization.

It is a way of formally representing knowledge such that the relationships between concepts is shown. Ontology has components to help describe a domain. A domain can be anything from products to medical information. A *Class* expresses a concept in the domain. Each element in the class is called an *Individual*. Properties of individuals or

characteristics are stored in *Attributes* under an individual. Last common component of ontology is called *Relation*. Relations tie two classes together for reasoning. Since contemporary ontologies share this components in the design, we can express information in different domains as ontologies.

**Definition (Domain) :** In order to do feature-based sentiment analysis, we first must define a domain. A domain can be anything from cameras to hotels, the size of the domain depends on the scope of the analysis we want to make. In this thesis we took TripAdvisor as a domain and modeled Comments and Hotels as classes in the ontology.

**Definition (Feature) :** Like the domain, the feature also depends on the requirements for the analysis. A camera's features can be lens, weight, battery, capacity whereas for a movie it can be artist, genre, acting, music and so on. There are no standard feature list for all domains. In our test dataset we define features of an hotel as business service, check in \front desk, cleanliness, location, rooms, service, and value.

**Definition (Polarity) :** From a sentiment analysis point of view, polarity can be positive or negative. One of the first research done on sentiment analysis was to find grammatical components, like adjectives, that carry polarity. The polarity of a word, or lexeme, can differ in a sentence. Take the word "rock" for example. In the sentence "There is a rock"; it does not contain any polarity. If you use it as a verb as in "This concert rocks", it contains a positive polarity. The project SentiWordNet focuses extensively on creating a database of words and their polarity in different usages.

**Definition (Sentiment) :** The sentiment as it stands in sentiment analysis, aims to find the attitude towards a certain concept. The sentiment can be calculated for a single sentence or a whole paragraph. In feature-based sentiment analysis, we aim to find the attitude towards certain features. Sentiment analysis is not restricted to finding out positive or negative attitude, it can also aim to find emotional states like sadness, or contempt. In this thesis, we restrain ourselves to finding if a feature has positive, negative or neutral sentiment.

**Definition (Scorecard) :** A scorecard is a summary of feature-sentiment pairs in a tabular way. It contains overall sentiment and overall sentiment score for a hotel as well as sentiments per feature. Sentiments are expressed as positive, negative or natural as well as an integer for easy comparison between two hotels or two features. Each feature also has a support value determined by the frequency of feature-related keywords found in the given dataset.

In this thesis we aim to find sentiment  $S$  towards a list of features  $F$  from a dataset of  $D$ . In  $D$ , we have comments written by real users, on hotels: expressed in an ontology. Our aim is to output the aggregated  $S$  for each  $F$  in the domain ontology. We summarize the relation between  $S$  and  $F$  into a scorecard with the help on NLP-techniques that will be detailed on the next chapter.

# Chapter 4

## Methodology

This chapter elaborates on our approach to the problem defined on Chapter 3. We'll first talk about evolution of our framework from a simple plugin to a full-fledged framework. Second, we'll give a top down look on our framework's architecture by dividing it into functional modules. In each module we'll highlight important concepts both old and novel. In the end, we aim to establish an understanding of the framework before moving on to experiments and results.

### 4.1 Evolution of Our Framework

During the research and development process we had three stages. We started with a pre-built framework; but as our research grew more complex, we improved our tools in order to satisfy our requirements. Below we will give a summary of each stage and capabilities of the framework.

#### 4.1.1 GATE

GATE (General Architecture for Text Engineering) is an open source project lead by Sheffield University [3]. It provides various tools on natural language processing, ontology, reasoning and data mining. While we were generating and testing our polarity words list, we used GATE's tools to mark polarity words on our test sentences.

#### 4.1.2 GATE+

After our initial research on polarity words list, we needed a tool to calculate polarity of a sentence using our word list and scoring scheme. Since we have the basic tools on GATE, we preferred to write a simple plugin that takes score of each word and adds them up. This plugin did simple lookup and calculation only.

### 4.1.3 Our Framework (Guinevere)

As our research went on we observed that looking up and scoring words may be enough to calculate polarity of a sentence up to a certain point; but it was not precise enough to place polarity on the right words. Polarity placement is crucial for our goal of feature-based sentiment analysis and we needed a special approach to place polarity in a sentence. Right at this time, we realized that we used only a small subset of tools GATE offered while sacrificing mobility and performance due to GATE's dependencies. Most of the tools we used in GATE were independent projects, so we decided to build our own modular framework while utilizing third party APIs for various tasks.

We utilize the following APIs:

- Stanford Core NLP API

We needed a capable natural language processing (NLP) toolbox to analyze comments. Stanford NLP package offered us a Part-Of-Speech Tagger [20] and a Statistical Parser [13, 2] to generate parse trees and dependency graphs.

- OWL API

OWL API [9] is an open-source project under GPL License that enables working with OWL ontologies. It handles our communication with domain, polarity and scorecard ontologies.

In the next section we'll give a top-down look of our architecture and modules that work together.

## 4.2 Architecture

With the goal of feature-based sentiment analysis in mind, we built our framework in a modular fashion so each part can be taken out, improved and put back into the framework. We also aimed our framework to be able to work with different domain ontologies. Here we have a look at our framework.

Our framework takes a paragraph, and first splits it to sentences. Then each sentence is further parsed with Stanford NLP API, to tokens. The NLP API also generates a parse tree and a dependency graph for given sentence. We then search our Polarity Ontology for tokens to get initial polarity values. After that our engine extracts noun phrases in the sentence using the parse tree. Finally, we parse through the dependency graph and using various rules we assign weight and final polarity values to each token. Another module named Sentiment Scorecard Generation Module prepares the final scorecard.

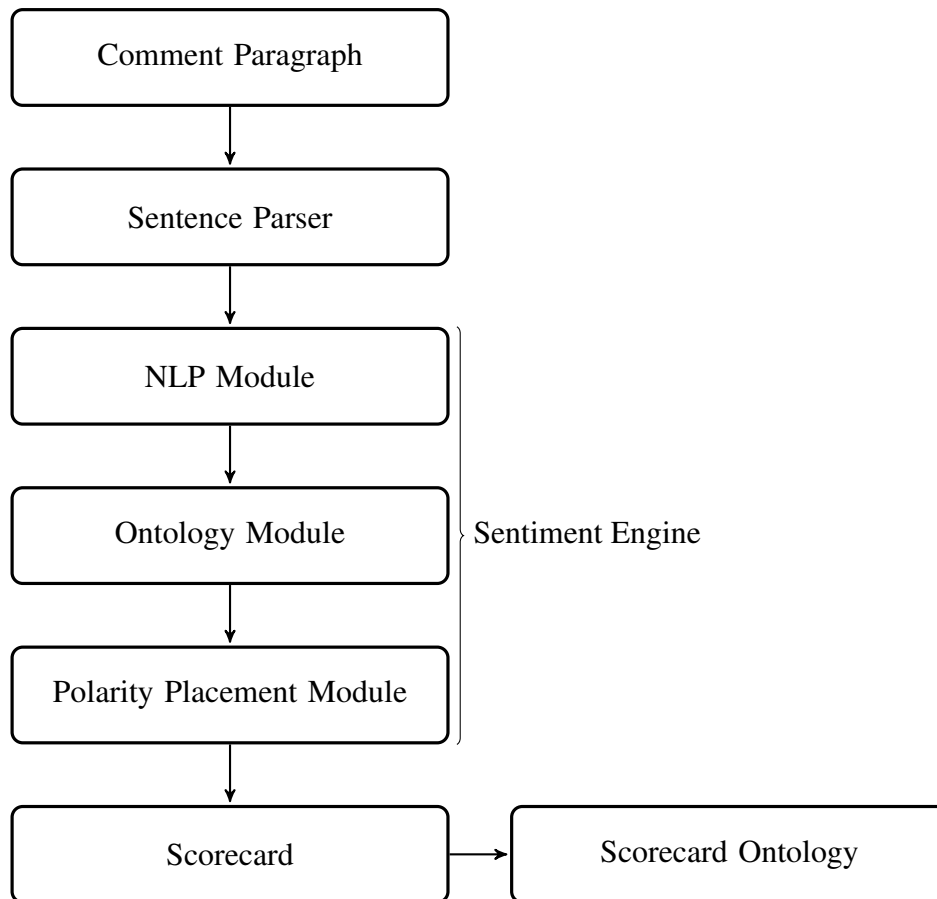


Figure 4.1: Overview of Our Framework

## 4.2.1 Data Structures

In order to store and analyze data we came up with three data structures

### 4.2.1.1 Parse Token

We have previously stated that we brake paragraphs into sentences, and sentences into tokens. There are many different information we need to keep on a token for analysis. In the table below we give a summary of all the attributes of a token in our framework.

Table 4.1: Attributes for Parse Token

Word	The token string without any modifications
Root	Morphological root of the word
POS Tag	Part-of-Speech Tag
Polarity	Polarity of the token kept as an integer
Implied Polarity	Polarity gained by dependency to another polar token
Weight	Strength of polarity of token in the given sentence
Ontology Class	If the token is also a keyword for a feature in given domain ontology, keeps the class information.



### 4.2.1.2 Parse Sentence

Parse Sentences are made up of tokens and polarity information. Table 4.2 shows attributes of a Parse Sentence.

Table 4.2: Attributes for Parse Sentence

Sentence	The sentence string without any modifications
Vector of Parse Token	Holds all tokens in the sentence
Parse Tree	Holds Parse Tree generated by Stanford NLP API in a Tree structure
Dependency Graph	Dependencies stored in a Semantic Graph data structure
Polarity	Polarity of the sentence stored as an enumeration (Positive, Negative, Neutral)
Polarity Score	Polarity of the sentence stored as an integer
Vector of Noun Phrases	Noun phrases extracted by the engine and stored in Parse Phrase structure.

### 4.2.1.3 Parse Phrase

Parse Phrases are first developed to store noun phrases in a sentence, but they can easily be extended to hold other phrases like idioms. Table 4.3 shows attributes of a Parse Phrase.

Table 4.3: Attributes for Parse Phrase

Vector of Parse Token	Holds all tokens in the phrase
Polarity	Polarity of the sentence stored as an enumeration (Positive, Negative, Neutral)

## 4.2.2 Modules

During our research and prototyping process we had utmost care for modularity. We divided our framework into three major modules: NLP module, Ontology Module and Sentiment Scorecard Generation Module. Each module has novel ideas for sentiment analysis and important aspects will be explained in the subsections.

### 4.2.2.1 NLP Module

This module helps us to better analyze a sentence and place polarity on correct token(s). First the sentence is passed through Stanford NLP component and resulting data are stored in data structures outline in Section 4.2.1, then we make analyze parse tree and dependency graph of a sentence. There are three important components in NLP module: POS Tags, Parse Tree and Dependency Analysis components.

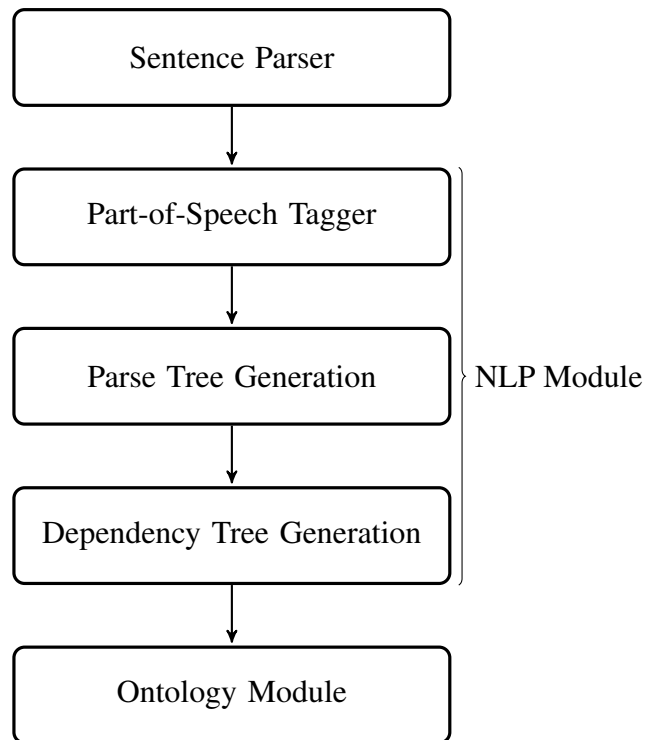


Figure 4.2: Overview of NLP Module

**4.2.2.1.1 Part-Of-Speech Tags** Part-Of-Speech tags shows the type and the form of a word in a sentence. An example of a POS Tagged sentence is given below.

*We just returned from four nights at the Hotel Monaco Seattle and absolutely loved it.*

we /PRP (0) just /RB (0) return /VBD (0) from /IN (0) four /CD (0) night /NNS (0) at /IN (0) the /DT (0) Hotel /NNP (0) Monaco /NNP (0) Seattle /NNP (0) and /CC (0) absolutely /RB (0) love /VBD (0) it /PRP (1) . / . (0) . / .

In a sentence words can be found in their normal, comparative or superlative forms. These forms are especially important for us since they determine the strength of the polarity in the sentence. As stated earlier we store polarity words in an ontology. If we generated comparative and superlative format and tried to store them in our ontology, we would generate an unnecessary clutter. In order to avoid this overhead we use two helpers. We pass token through a Morphological Analyzer to get the lexeme for polarity lookup and then we look at the POS tag of the word in the sentence to determine if it is in comparative or superlative form. If the word is in one of the two forms, then we adjust the weight of the token accordingly. Weight will be doubled for comparative form and tripled for superlative form. POS Tags that end with *R* denotes a word in comparative form, whereas a tag ending with *S* superlative form.

There is one other benefit of analyzing POS Tags in a sentence; we can determine the type of a given word. In English language a word can be used interchangeably in different

type. Take the word *rock*; it can be used as a noun as in “*Here is a rock*” or as a verb “*Waves are rocking the boat*”. We divided our polarity words in four categories by their types: verb, adjective, noun and adverb. Using POS tags enable us to correctly lookup the polarity of a word.

**4.2.2.1.2 Parse Tree** Parse Trees give us the grammatical structure of a sentence. We leveraged parse trees obtained to find and extract noun phrases in a given sentence for further analysis. In our system parse trees can either be stored in a Tree data structure or in a condensed string format. The condensed format shows grammatical structures in an easier form.

*We just returned from four nights at the Hotel Monaco Seattle and absolutely loved it.*

```
(ROOT (S (NP (PRP We)) (ADVP (RB just)) (VP (VP (VBD returned) (PP (IN from)
(NP (NP (CD four) (NNS nights)) (PP (IN at) (NP (DT the) (NNP Hotel) (NNP Monaco)
(NNP Seattle)))))) (CC and) (VP (ADVP (RB absolutely)) (VBD loved) (NP (PRP it))))
(. .)))
```

In the condensed format, *NP* denotes that string between parenthesis is a noun phrase. This way we can go through the parse tree to gather noun phrases for further analysis. Also we can gather other phrases like idioms or slang phrases for a more fine-grained analysis.

**4.2.2.1.3 Dependency Analysis** Dependency analysis allows us to take a closer look at grammatical relationship between two words in a sentence. Using the dependency analysis, we can find out modifiers or negators and take them into account while determining the sentiment for a feature. Below an example of dependency analysis on an example sentence is given:

We just returned from four nights at the Hotel Monaco Seattle and absolutely loved it.

```
subj(returned-3,We-1)
nsubj(loved-14,We-1)
advmod(returned-3,just-2)
num(nights-6,four-5)
prep_from(returned-3,nights-6)
det(Seattle-11,the-8)
nn(Seattle-11,Hotel-9)
nn(Seattle-11,Monaco-10)
prep_at(nights-6,Seattle-11)
advmod(loved-14,absolutely-13)
conj_and(returned-3,loved-14)
dobj(loved-14,it-15)
```

We will be defining certain rules using dependency analysis to place polarity on the correct word in a sentence. Our approach will be elaborated in section 4.2.3. Even though feature extraction is beyond the scope of this thesis; dependency analysis is used by [8] to define rules for extracting opinion words from a corpus.

#### 4.2.2.2 Ontology Module

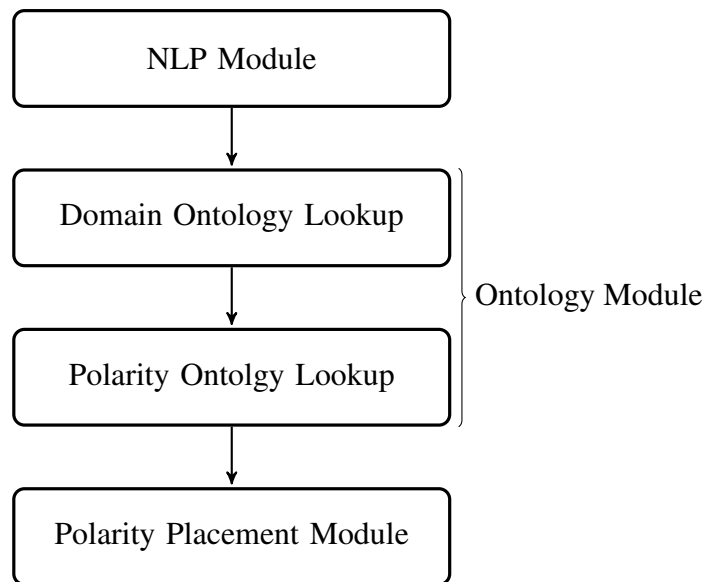


Figure 4.3: Overview of Ontology Module

As stated before, ontology is used to define concepts in a domain in a formal way. Our framework uses three ontologies: one for representing domain and feature knowledge; one for representing polarity information for lexemes and one for feature-based analysis summary and the logic behind it. We will briefly describe important aspects of each ontology below. Figure 4.1 shows the overview of our test ontology with relations between each class.

- Domain Ontology** Each domain ontology will have a different structure to conceptualize the information. In our Tripadvisor ontology, we have two main classes: Hotels and Comments. Each hotel can have infinite comments, and a relation is defined between them. Since feature-extraction is an extensively search topic, we assumed that we have the features and feature related keywords available prior to analysis. We extended Tripadvisor ontology by adding another class Features and seven sub-classes under it. Each subclass represents an aspect of the hotel. The seven subclasses are Business Service, Check In \Front Desk, Cleanliness, Location, Rooms, Service, Value.

Under each subclass we have keywords related to respective feature as individuals. Each individual have two data properties: *feature\_polarity* and *feature\_weight*.

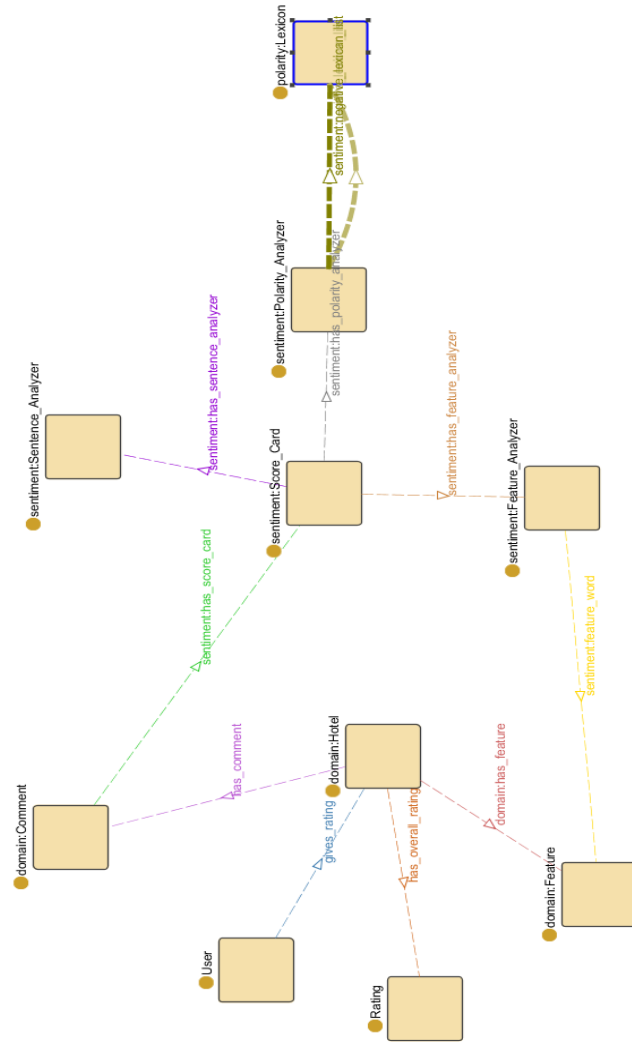


Figure 4.4: Overview of Our Ontology Structure

The reason behind such a structure is that analyzing polarity of the words using a database approach will not always yield correct result due to differences in the domain. Consider the word "fast" as an example. In a camera ontology, "fast" has a positive polarity if the word is related to shutter speed. On the other hand, if "fast" is related to battery life; it has a negative polarity. While we do not propose an automatic system, we manually polarities of features in our domain ontology. This approach allowed us to take domain-specific knowledge into account while placing polarities.

Same as polarity, we need to have a way of determining keywords that are more important for sentiment analysis. The criteria for weight assignment is related to the preferences of the analyst and completely subjective. Defining weights in the structure also allows us to work with profile based systems, where different profiles may define different weights for keywords. We leave any research in this direction as future work.

Any domain ontology extended using a Feature class and necessary subclasses can be used with our framework, giving it flexibility.

- **Polarity Ontology** There are two reasons for expressing our polarity keyword as an ontology: 1) To easily extend our polarity knowledge with new classes for phrases, idioms or other slang words. 2) To define relations with the Scorecard ontology for aggregating the logic behind the output. For this thesis we defined one main class. Lexicon and four subclasses adj, adv, noun and verb. The reason behind this structure to take into account, different usages of a word and different polarities it can get.

Under each class we have keywords as individuals associated with their respective classes. Each individual has 4 attribute: pos\_value, obj\_value, neg\_value and value. The first three are gathered from average of SentiWordNet results. We further classified each word as positive or negative by applying a threshold and obtained a single integer for polarity value. We take this integer value as basis on our analysis.

- **Scorecard Ontology** One of the aims of this thesis is to summarize the result of feature-based analysis in an easy to read and compare form. We call this summarized result a scorecard. Scorecard ontology holds all the data that is necessary for scorecard creation. It has four classes; *score\_card*, *feature\_analyzer*, *polarity\_analyzer* and *sentence\_analyzer*. Class *score\_card* has an overall\_value as an integer, two boolean fields (isPositive and isNegative) and has relations to other classes. We will give a summary of each related class below:

- **Feature Analyzer** stores information about each feature found in a comment on separate instances. An instance holds *overall value*, *support value* and *set*

*of feature-words* our engine found while processing a comment.

- **Polarity Analyzer** stores *sets of positive and negative lexicons* our engine found during processing.
- **Sentence Analyzer** stores *number of positive and negative sentences* in a scorecard.

A `score_card` holds one `polarity_analyzer`, one `sentence_analyzer` and `n` `feature_analyzers` with `n` being the number of features found in the analysis.

Using the information stored in the scorecard ontology, we can search for different features and we will be giving example test cases on Section 5.5 without rerunning the NLP component.

### 4.3 Polarity Placement

As we mentioned before, Feature-based sentiment analysis requires a higher level of precision than document level sentiment analysis. While working on the document level, finding the overall sentiment is the main goal. With feature-based approach; we need to be able to place polarity on the correct word. In this case we need to place the polarity on the feature keywords so that we can generate a correct scorecard. In order to achieve this goal we need to analyze the given document (in our test case user comments). A step by step description of our system is given below.

- Comments are separated into sentences. A Vector of sentences are obtained. These sentences are converted to Parse Sentence data structure.
- These sentences are parsed one by one using NLP module and a vector of Parse Tokens are generated. During the parsing process, POS Tags and Dependency Graph are also obtained and stored for further analysis.
- Using Ontology module, we search the polarity ontology for each Parse Token. In this search we use previously generated POS Tags to identify the form of the word in the sentence. If a token is in one of *adjective, adverb, noun or verb* forms, we search the respective class for polarity information. Every Parse Token starts with *weight* as 1.
- Using Ontology module, we search the domain ontology's feature class for each Parse Token. If the token matches with a feature keyword in the ontology, we update three attributes of the token: *ontology class, polarity and weight*.
- Our sentiment analysis engine first checks POS tags for any comparative or superlative usage. A comparative use increases weight by 1; and a superlative usage increases weight by 2.

- With all the polarity and weight information in place; our engine starts to go over the dependency tree one by one to transfer polarity between Parse Tokens. The rules and their effects are given in Section 4.3.1.
- After all the dependencies are resolved and polarity and weight values are assigned, the sentiment analysis is complete. Scorecard generation is done by parsing each Parse Sentence and Parse Token to aggregate sentiment to features.

### 4.3.1 Dependency Rules

We use dependency graph we obtained from NLP module and analyzer certain dependencies to adjust *polarity*, *weight* and *implied polarity* attributes of a Parse Token. We will group certain dependencies together. In order to mathematically show the transfer of polarity, we express negative sentiments with negative integers and positive sentiments with positive integers. Zero polarity means that the token does not hold any polarity. Weights are always expressed by positive integers starting with one.

#### 4.3.1.1 Dependencies That have a Direct Impact

In this subsection we will discuss handling of dependencies that have a direct impact on the polarity of head token. This section focuses on the following dependencies:

- Adverbial Modifier (advmod)
- Adjectival Modifier (amod)
- Noun Phrase as Adverbial Modifier (npadvmod)
- Prepositional Clausal Modifier (prepc)
- Open Clausal Complement (xcomp)

Our system transfers polarity between token or increases the weight for this dependencies. We will show the effects on a case by case basis:

- **Case 1**  
if

$$\begin{aligned} \text{word1}_{\text{polarity}} &= \text{word2}_{\text{polarity}} \\ \text{word1}_{\text{polarity}} &\neq 0 \end{aligned}$$

then

$$\text{word1}_{\text{weight}} = \text{word1}_{\text{weight}} + \text{word2}_{\text{weight}}$$



- **Case 2**

if

$$\begin{aligned} \text{word1}_{\text{polarity}} &= \text{word2}_{\text{polarity}} \\ \text{word2}_{\text{polarity}} &< 0 \end{aligned}$$

then

$$\text{word1}_{\text{polarity}} = \neg \text{word1}_{\text{polarity}}$$

if

$$\text{word1}_{\text{polarity}} = \text{word2}_{\text{polarity}} \text{word2}_{\text{polarity}} > 0$$

then

$$\text{word1}_{\text{weight}} = \text{word1}_{\text{weight}} + \text{word2}_{\text{weight}}$$

- **Case 3**

if

$$\begin{aligned} \text{word1}_{\text{polarity}} &= 0 \\ \text{word2}_{\text{polarity}} &\neq 0 \end{aligned}$$

then

$$\text{word1}_{\text{polarity}} = \text{word2}_{\text{polarity}}$$

#### **4.3.1.2 Negators**

We resolve any negator dependency by doing

$$\text{word1}_{\text{polarity}} = \neg \text{word1}_{\text{polarity}}$$

#### **4.3.2 Object of Partial Modifiers**

The difference between direct and partial modifiers is that, partial modifiers will only modify the polarity of a word but not the weight. Also the transfer direction of the polarity is determined by the weights of Parse Tokens. The token with smaller weights loses its polarity. We handle these dependencies under this section:

- Direct Object (dobj)
- Participial Modifier (partmod)
- Relative Clause Modifier (rmod)
- Relative (rel)
- Dependent (dep)

We can express the operation as  
if

$$\begin{aligned} \text{word1}_{\text{polarity}} &\neq 0 \\ \text{word1}_{\text{polarity}} &> 0 \\ \text{word2}_{\text{polarity}} &< 0 \end{aligned}$$

and if

$$\text{word1}_{\text{weight}} > \text{word2}_{\text{weight}}$$

then

$$\text{word2}_{\text{polarity}} = \text{word1}_{\text{polarity}}$$

else

$$\text{word1}_{\text{polarity}} = 0$$

if

$$\begin{aligned} \text{word1}_{\text{polarity}} &\neq 0 \\ \text{word1}_{\text{polarity}} &> 0 \\ \text{word2}_{\text{polarity}} &= 0 \end{aligned}$$

then

$$\begin{aligned} \text{word2}_{\text{polarity}} &= \text{word1}_{\text{polarity}} \\ \text{word1}_{\text{polarity}} &= 0 \end{aligned}$$

### 4.3.3 Noun Dependencies

We also looked at dependencies between nouns by analysing Noun Compound Modifiers (nn) and Nominal Subject (nsubj) dependencies. The main difference with these types of dependencies is that, we do not transfer the polarity or weight; but we rather update *implied polarity* attribute. The difference between polarity and implied polarity is that implied polarity is not taken into account in final sentiment calculation but used for feature-based sentiment summary. If one of the words have polarity, the polarity of one word is set as implied polarity of the other word.

After polarity placement, the last step for the framework is to feed necessary data for scorecard generation to scorecard ontology. We will give detailed information on scorecard representation in the next section.

## 4.4 Scorecard Representation

Previous sections elaborate on our approach to finding feature-sentiment pairs. In this section, we approach to another aspect of the problem - summarizing the results in an easy to comprehend way. During our research, our focus was on the features in a domain. In summarizing the results, we retain the same focus. Scorecard is a tabular structure that aims to show all feature-sentiment paris and supporting information in a single page. Table 4.4 gives an example scorecard produced by our framework using Tripadvisor ontology mentioned earlier.

Hotel 72572			
Overall Sentiment Score 866			
Aspect	Overall Sentiment	Overall Sentiment Score	# of Keyword Appearance
Check In \ Front Desk	POSITIVE	47	92
Service	MORE POSITIVE	27	41
Rooms	MOST NEGATIVE	-30	31
Value	POSITIVE	4	17
Location	NEUTRAL	7	91
Business Service	NEUTRAL	0	2
Cleanliness	NEGATIVE	-1	4

Table 4.4: A Sample Scorecard Produced by Our System

This layout allows a user to see the summary of the analysis in a single and clean table. *Overall Sentiment Value* and *# of Keyword Appearance* are present in the scorecard to enable comparison between two scorecard. With this representation we were able to compare two hotels in the ontology using their scorecards. A more detail example will be given on Section 5.5.

Even though from a user perspective, this scorecard summary is enough: an analyst may want to have a more detailed look at the results. We also give another tabular summary called Scorecard Log. This log contains detailed information on the base results that are aggregated. An example of a Scorecard Log is given on Table 4.5 . Using this log an analyst can see which feature keywords contributed more to feature detection and sentiment.

Hotel 72572		
Comment_23		
Overall Sentiment: POSITIVE		
Comment Sentiment Score : 46		
Sentence Analyzer		
Number of Positive Sentences : 10		
Number of Negative Sentences : 3		
Polarity Analyzer		
Keyword	Polarity	Weight
best	3	1
hotel	1	2
Feature Analyzer		
Check In \Front Desk		
Overall Sentiment: POSITIVE		
Sentence Sentiment Score : 2		
Support : 3		
Keyword	Polarity	Implied Polarity
staff	0	1
friendly	1	0
helpful	1	0
.		
.		
.		

Table 4.5: A Sample Scorecard Log Produced by Our System

## 4.5 Finding Overall Sentiment

Our framework aims to present the results in an user-friendly way with a scorecard representation. Since feature-based sentiment analysis is fine-grained, our results must also be fine grained. In order to compare between objects or aspects; we crated our own scale for Overall Sentiment. In our scale there are 7 sentiments from Most Negative to Most Positive. Table 4.6 will give the criteria for each sentiment rating.

Since our system already gives us two metrics *Overall Sentiment Value* and *# of Keyword Appearance*. In a scenario where each keyword about an aspect is positive; *Overall Sentiment Value* must be equal to or bigger than *# of Keyword Appearance*. In real test

cases, there are positive and negative sentiments about an aspect in a set of comments; so *Overall Sentiment Value* will always be smaller.

In order to make comparisons between hotels with different comment sizes, we need to put them into same scale. If we divide *Overall Sentiment Value* by *# of Keyword Appearance*; we have a floating point number between -1 and 1. We will refer to this number as *Rating* or *R*. Defining Neutral sentiment was also important, since taking only  $R = 0$  would restrict neutrality to a single value. We gave Neutral sentiment value a range between - 0.1 and 0.1 in order to make our scale more reliable. For the highest level of polarity, we took top 0.1 range. The rest is divided into to two equal ranges.

Table 4.4 also shows our rating scale embedded into the sample scorecard. This scale allows a user an easy way to compare two feature or two hotels. In next chapter we will define performance metrics, give performance test results.

Overall Sentiment	Scale
Most Negative	$R \leq - 0.9$
More Negative	$- 0.9 < R \leq - 0.5$
Negative	$- 0.5 < R < - 0.1$
Neutral	$- 0.1 \leq R \leq 0.1$
Positive	$0.1 < R \leq 0.5$
More Positive	$0.5 < R < 0.9$
Most Positive	$0.9 < R$

Table 4.6: Table of Ratings

# Chapter 5

## Experiments

### 5.1 Test Data Set

In order to evaluate our framework we used TripAdvisor dataset from The University of Illinois at Urbana-Champaign, The Database and Information Systems Laboratory <sup>1</sup>. There are 1805 hotels and 108891 comments in this dataset. This dataset also contains 7 aspects and their respective ratings for each comment, along with a seed list of keywords related to these aspects.

The general evaluation strategy in sentiment analysis is to randomly select comments from IMDB or a product site (like Amazon); and then have human labelers to label these comments as positive or negative [16, 26, 23]. This approach arises two problems: 1) Although some research groups share their randomly selected datasets, many groups do not. In order to compare two different approaches, we need to use the same set of comments. This is impossible, unless the research group publishes their set. 2) The cost of human labor to manually label comments is expensive. In TripAdvisor dataset, there average sentence per comments is around 8. This cost prohibits groups from working large sets. Many published papers use a small set of 100 comments [16, 26]. In the Performance Evaluation Section we will give a comparison of results from two different sized datasets, and show that the smaller set gives out better results.

Even though TripAdvisor set is generally used on evaluation of recommender systems [15, 22]; these problems stated above motivated us to use it for performance tests. The set addresses both problems: 1) The data set is available on the internet, and it will be available to other groups working on sentiments analysis. 2) The dataset contains a user-given rating for each aspect on each comment. This enables us to take user-ratings as base truth and skip manually annotating the comments. It enabled us to work with larger sample sets for evaluation.

For the experiments we modeled TripAdvisor domain as an ontology as we mentioned

---

<sup>1</sup>The dataset is available on <http://sifaka.cs.uiuc.edu/wang296/Data/index.html>

on previous chapters. We selected 500 hotels along with 74202 comments associated with them. In our domain we have 7 features and total of 309 feature words associated with them. In our evaluation marked user-ratings with Rating 1 as Negative; and Rating 5 as Positive.

### 5.1.1 Test Data Integrity

User ratings are taken as ground truth in our experiments, and we evaluate our performance by comparing our system’s result with user ratings. For each comment we have 7 separate aspect ratings and an overall rating. We put the correlation between aspect ratings and overall rating in our test dataset to check against any bias. Intuitively, one can expect a user giving a 1-star overall rating to give 1-star to each separate aspect; since 1 star denotes a very negative rating. Similarly a user giving 5-star rating in a comment should give high ratings (4 or 5 stars) to each aspect.

In order to check integrity, we looked at three different ratings. We mapped out the distribution of ratings for 1, 3 and 5 star ratings. The Figures 5.1, 5.2 and 5.3 shows the results.

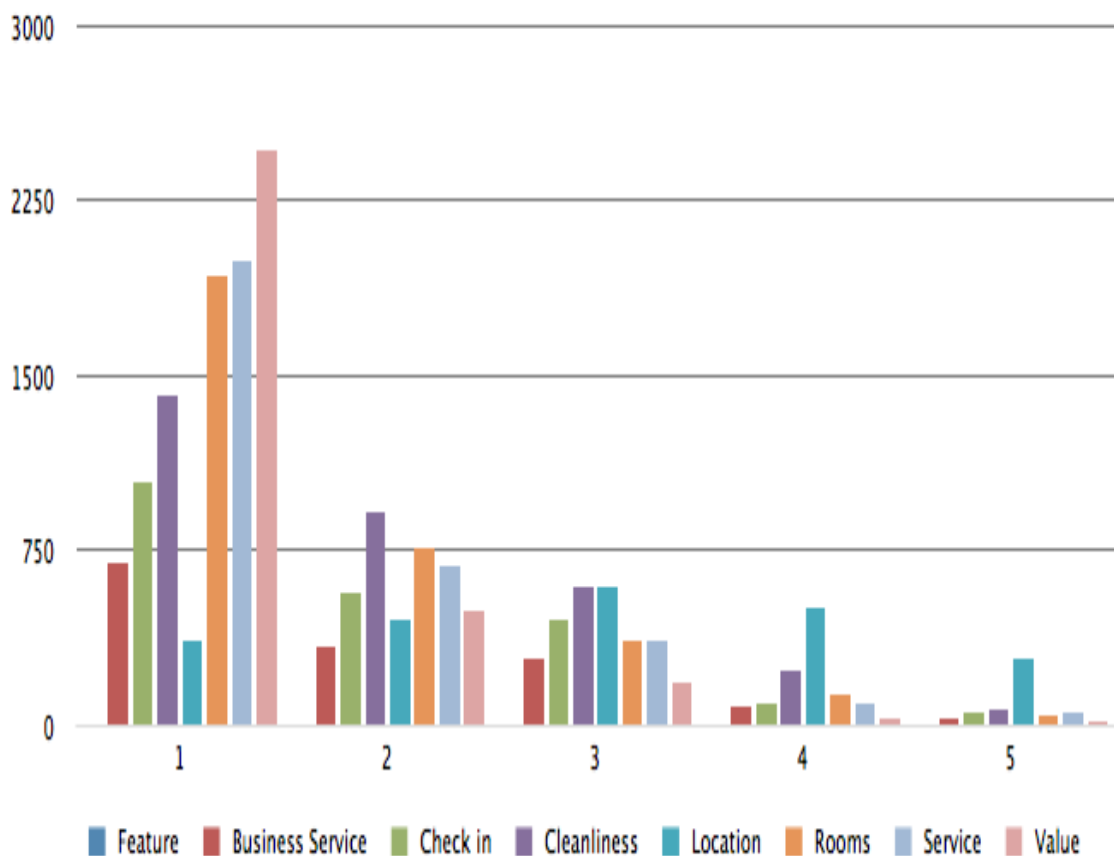


Figure 5.1: Distribution of Aspect Ratings for Overall Rating 1. X-axis shows star-ratings and y-axis shows number of comments.

As can be seen from Figure 5.1; most users gave 1-star rating to other aspects when they give 1-star rating on overall rating. 2-star ratings came second and there are a small group giving ratings 3 or more to aspects. This shows an inconsistency in user behavior; a user giving a low rating on overall does not necessarily give a low rating on aspect-basis.

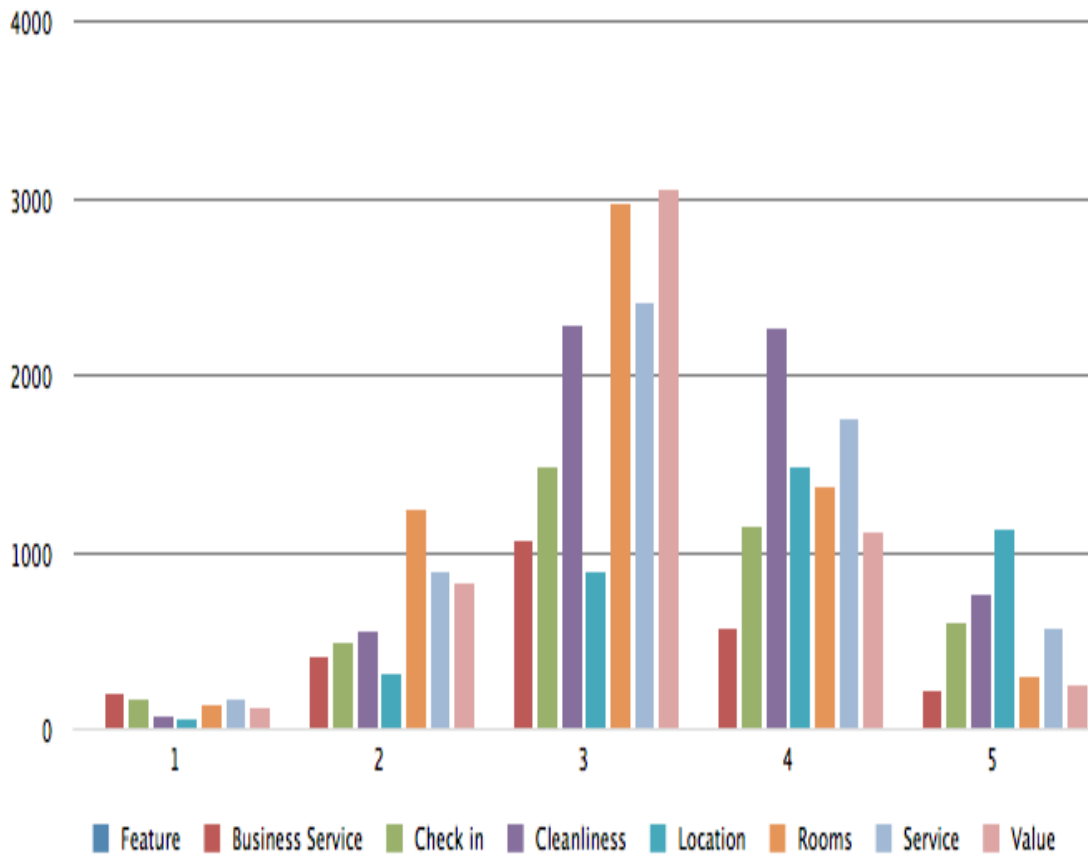


Figure 5.2: Distribution of Aspect Ratings for Overall Rating 3. X-axis shows star-ratings and y-axis shows number of comments.

Figure 5.2 shows distribution for Overall Rating 3. As expected this is a normal distribution with 3-star ratings having the highest number of comments. We see a smaller number of comments on 1-star ratings than 5-star ratings, meaning that dissatisfied users tend to give lower ratings to aspects. If a user gives an overall rating 3, and we take it as neutral sentiment, users generally rate other aspects between 2-4 stars.

Figure 5.3 shows that when users give a 5-star rating on overall, they are very pleased with the other aspects too. Contrary to Overall Rating 1 (Figure 5.1), there number of comments with 1, 2 and 3-star ratings are very small. In next section, we define performance metrics for our system.



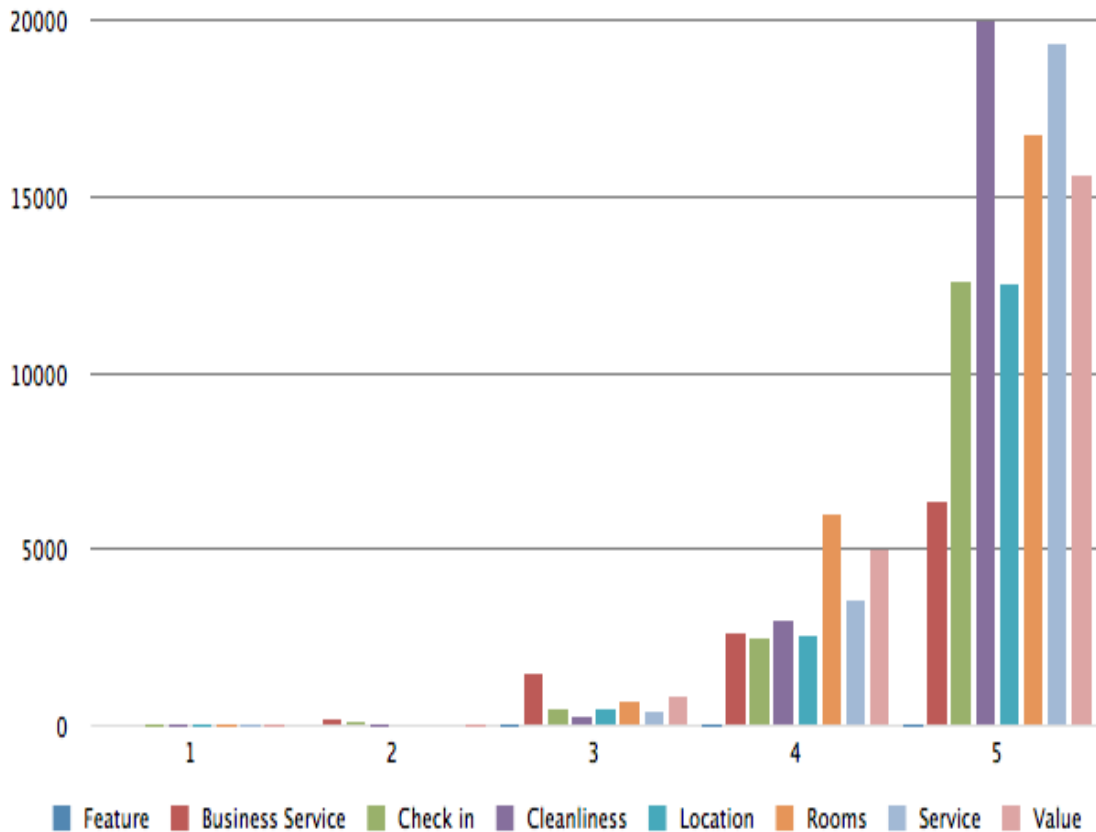


Figure 5.3: Distribution of Aspect Ratings for Overall Rating 5. X-axis shows star-ratings and y-axis shows number of comments.

## 5.2 Performance Metrics

We have three metrics defined to evaluate the performance:

- Precision

In our test environment we define precision as ratio of total number of correctly assigned sentiments against total number of correct and misplaced sentiments in our test dataset.

**Example** Precision for positive comments in a test dataset is

$$\frac{True_{positive}}{True_{positive} + False_{positive}} \quad (5.1)$$

- Recall

In our test environment we define recall as ratio of correctly assigned sentiments against total number of known sentiments in our test dataset.

**Example** Recall for positive comments in a test dataset is

$$\frac{True_{positive}}{True_{positive} + False_{negative}} \quad (5.2)$$

- F-measure F-measure is the harmonic mean of precision and recall.

$$2 * \frac{precision * recall}{precision + recall} \quad (5.3)$$

### 5.3 Performance Evaluation

In order to calculate our performance metrics, we selected 500 comments with rating 1 to represent negative comments and 500 comments with rating 5 to represent positive comments for each aspect. These comments are selected by generating the list satisfying the test criteria (Ex: Comments with Value aspect rating 1), shuffling the list three times and finally taking top 500 comments on the list.

Our first experiment was to evaluate the performance with all modules enabled for each feature in our test ontology. For each feature we generated a subset for both negative and positive comments. Table 5.1 shows the results.

Feature	Sentiment	Precision	Recall	F-Measure
Business Service	Positive	0.695	0.808	0.747
	Negative	0.89	0.508	0.647
Check In \ Front Desk	Positive	0.749	0.858	0.800
	Negative	0.923	0.552	0.691
Cleanliness	Positive	0.781	0.756	0.768
	Negative	0.933	0.612	0.739
Location	Positive	0.730	0.766	0.747
	Negative	0.809	0.534	0.644
Rooms	Positive	0.754	0.748	0.751
	Negative	0.930	0.562	0.700
Service	Positive	0.749	0.764	0.756
	Negative	0.933	0.582	0.717
Value	Positive	0.766	0.832	0.798
	Negative	0.921	0.630	0.748
Results Average	Positive	0.746	0.790	0.767
	Negative	0.906	0.569	0.698

Table 5.1: Evaluation Results for Aspects

After this experiment, we wanted to take comments with ratings 2 and 4-stars into our test-set. So we randomly prepared a test dataset with 75 comments with 1-star, 75 comments with 2-star, 75 comments with 4-star and 75 comments with 5-star rating. In this experiment we considered 1 and 2-star ratings as negative; 4 and 5-star ratings as positive. Table 5.2 shows the performance evaluation for this dataset.

Feature	Sentiment	Precision	Recall	F-Measure
Business Service	Positive	0.652	0.8	0.719
	Negative	0.845	0.327	0.471
Check In \ Front Desk	Positive	0.656	0.853	0.742
	Negative	0.947	0.45	0.610
Cleanliness	Positive	0.734	0.782	0.758
	Negative	0.857	0.52	0.647
Location	Positive	0.666	0.733	0.698
	Negative	0.782	0.48	0.595
Rooms	Positive	0.662	0.693	0.677
	Negative	0.869	0.486	0.624
Service	Positive	0.660	0.71	0.68
	Negative	0.86	0.466	0.606
Value	Positive	0.758	0.881	0.815
	Negative	0.914	0.573	0.705
Results Average	Positive	0.684	0.779	0.727
	Negative	0.869	0.471	0.608

Table 5.2: Evaluation Results with Different Ratings

We also wanted to investigate the impact of feature-keywords set on our analysis results. In order to evaluate this, we first generated 2 sets for each feature in our test ontology with 200 comments with negative and positive sentiments. Table 5.3 shows the accuracy of feature detection on given sets.

Feature	# of Feature Keywords	Sentiment	# of Comments with Keyword	Accuracy (%)
Business Service	46	Positive	83	33.2
		Negative	53	21.2
Check In \ Front Desk	39	Positive	186	74.4
		Negative	157	62.8
Cleanliness	16	Positive	115	46
		Negative	73	29.2
Location	52	Positive	137	54.8
		Negative	167	66.8
Rooms	65	Positive	180	72
		Negative	174	69.6
Service	61	Positive	168	67.2
		Negative	156	62.4
Value	30	Positive	146	58.4
		Negative	115	46
Results Average	309	Positive		58
		Negative		51

Table 5.3: Detection Accuracy for Features

## 5.4 Performance Discussion

Our first experiment evaluated the performance of our system with regards to features in our test set. If you look at the average of the results, our framework works with %75

precision for positive sentiments, and %90 for negative sentiments. On recall, positive sentiments are %79 where negative sentiments are %57. We investigated the difference between positive and negative comments by manually checking the results. The investigation pointed to one aspect of sentiment analysis we omitted in this thesis: implicit and comparative sentences. Comments like *"Food could be better in the breakfast"* or *"The room is smaller than a closet"* are misinterpreted by our system. We also found out that users generally used comparative sentences when writing negative reviews on hotels. When praising an aspect or writing positive comments, people tend to do it more explicitly. This misinterpretation of negative sentiment hinders recall rate of negative sentiments and precision of positive sentiments. Since the number of implicit expressions are considerably smaller in positive comments we have a high precision rate for negative review identification. Since it is a research filed on it's own [6, 4], we left sentiment analysis of comparative sentences as future work.

Even though we do not use the same test dataset with [10, 26, 23] we still have marginally higher precision than all of them. This is because we put effort into simplifying SentiWordNet data as well as our engine's analysis of dependency tree in sentence. This way we were able to correctly adjust the strength of the sentiment word as well as placing polarity on correct feature. We also used domain-specific polarity information to improve our precision. In order to increase the recall performance of our framework, we must incorporate implicit and comparative opinion analysis as a separate module. This addition will also increase the precision of negative sentiments.

Our second experiment showed that there is no linear correlation between number of feature-keywords and accuracy of feature detection in a comment. When we manually looked at the test set, we saw that there are two reasons for this. 1) People use different words to express the same sentiment; so in cases where we were unable to capture the sentiment users actually used keywords that are not in our seed list. 2) Sometimes users rate a feature, but do not mention anything about it in the review. This case is more apparent in negative reviews, where a customer dissatisfied with one or more feature may downrate all other features of a hotel without actually having negative sentiments towards it. The results also show that selection of feature-keywords is an important parameter to the quality of feature-based analysis.

Next section will give some use cases where results obtained from our framework can be used in real-life scenarios.

## 5.5 Use Cases

Summarizing sentiments in a scorecard also allows us to give personalized results based on requirements. Since each scorecard holds the summary for each feature separately; we can search our test domain ontology and gather results based on sentiments. We will give

three cases where our framework can be used effectively.

- The first case deals with finding out top-k results for given feature. Most product sites as we as Tripadvisor already have such a filtering system in place. However, most of the systems require a separate input of rating as integers or stars; meaning they require explicit information from the user.

From our test data-set we can find top-k hotels for a given feature and sentiment. For example, we can search for a hotel with the best Service, or the worst Location from given ontology. As we mentioned before a similar system is already in place on Tripadvisor; but our system can extend their filtering system by introducing new features and mining sentiments based on those features. Tripadvisor currently have filters for only four aspects (Value, Cleanliness, Service, Sleep Quality) where our framework can produce scorecards for seven aspects and it can easily be extended.

- Another use case is to find the dominant feature amongst a set of features. Nearly all news sites have a comment box under each news item. In such a case, scorecard-based approach would enable us to rank the sentiments by analyzing the comments left. Example: You are a media advisor for a presidential campaign and want to find out about issues that public has strong sentiments towards. Using our framework and a news domain ontology; you can identify the likely issues as features; then parse comments taken from news site and generate scorecards for each news comment. Querying for the feature with the strongest support will give you the most strong sentiment without any explicit information from commenters.

If we return to our test-set; analyzing the scorecards will give us the feature with the most support. This way we can find out the aspect that hotel reviewers most commented on. Our analysis revealed that for our test dataset Rooms was the most commented on aspect for both positive and negative sentiment sets with 2079 and 1679 support. In other words in 1000 comments people referred to Rooms aspect a total of 3758 times.

- Our scorecard based approach can also be used to compare two hotels in our test domain. You can compare them based on their overall sentiment value, or based on individual features. Example: You are searching for the hotel with good Service. You first select the hotels with five star Service ratings (as Tripadvisor stores ratings as five star), but that is as far as Tripadvisor's or other product sites filters get you. In order to decide between hotels with five star Service, you need to go over the comments and use your best judgement. Our scorecard approach also stores a quantified value for sentiment as *Sentiment Score and Support*. Table 4.4 gives out an example scorecard. *Sentiment Score* quantifies the polarity of the opinions,

where *# of Keyword Appearance* gives the frequency of the feature mentioned on selected hotel's comments. Using these two values you can compare two hotels with the same rating.

- Our scorecards also enable to do complex querying based on different feature sets. Take a family with a teenage child as an example. For the parents, Location and Value features are more important; where for the child Service aspect is more important. In order to retrieve the list of hotels satisfying all the criteria, we can first generate a list of hotels for parent's requirements and then we can generate a list for the child's requirements. In the end, the intersection of two lists will give the list of suitable hotels for this family. Since our feature-list is dynamic; it can easily be extended to handle complex requirements and allow users to do a more fine-grained search based on user comments.

As we mentioned above, finding and summarizing sentiments in a scorecard allows it to be used in many different scenarios. One last point of inquiry is the reliability of user comments for analysis. With the emergence of opinion mining systems, the problem of spam opinion detection also emerged [11, 12]. However, it is still harder to spam a comment-based analysis than a rating based analysis; due to nature of the problem. Malwares that spam rating and voting based systems are around for quite a while. In order to make an impact on comment based analysis, comments with correct grammar and keywords must be generated and posted in a contextually coherent way to make an impact. However hard, it is still not impossible, and researcher is done on this aspect of sentiment analysis.

In conclusion, we will give a brief summary of novel approaches in this thesis as well as some future works that can be done to enhance the performance metrics of our framework.

# Chapter 6

## Conclusion and Future Works

Our contributions in this thesis can be summarized as:

- We proposed a novel way of polarity placement by analyzing the dependencies between words to transfer polarity from polarity keywords to feature keywords in a sentence.
- We proposed a polarity ontology containing approximately 20,000 words classified into adverb, adjective, noun and verb categories. The words are both labeled by SentiWordNet system and author of this thesis for accuracy. Our polarity ontology can easily be extended to hold phrases, idioms and slang words for increased precision and recall.
- We proposed a scorecard structure and accompanying ontology for effective storage and querying. This structure allows our framework to be easily adapted to many different scenarios as mentioned in Section 5.4 Use cases. Scorecard structure also has its own user-friendly rating system that scales with text size.
- Our framework is made up of modules and can be easily extended to accompany research mentioned in Future Works section.

While we achieved our aim to propose a working feature-based sentiment analysis framework, there are some topics we feel will benefit the performance or extend the functionality of our framework. Future works may include:

- **Profile-Based Aspect Querying:** As mentioned before, our scorecard summaries allow users to get results based on their preferences. Our framework also support different weight and polarity values for feature-identifying keywords. A profile-based aspect-based system can leverage this functionality to generate different feature classes suitable for different profiles. For example, a teenage profile may give more weight to location and value, where an adult profile may increase the weight of service aspect. This way our framework can be extended to other research tracks.

- **Sentiment-Based Recommender Systems:** Recommender systems use values that are quantifiable or classifiable as input values. Our framework quantifies sentiment by scoring and giving support value to our analysis. This was a recommender system and we use our framework to recommend hotels or products based on features that people expressed positive sentiments.
- **Feature Extraction:** The number of feature-keywords under feature class has an impact on the quality of the analysis. The bigger the feature-keyword set gets, the more fine-grained the results will be. As mentioned on Chapter 2, feature-extraction is an extensively topic with focus on machine learning algorithms. We feel that addition of a feature extraction module will allow autonomous expansion of feature class and improve our performance results.

This thesis aims to propose a framework for feature-based sentiment analysis while focusing on polarity placement and sentiment summarization. The framework is modular and easily extendable; and designed to be the core component of UBIPOL sentiment analysis module.



## Bibliography

- [1] 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 (LREC10)*, Valletta, Malta, May. European Language Resources Association (ELRA). Citeseer, 2010.
- [2] Marie catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *In LREC 2006*, 2006.
- [3] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002.
- [4] X. Ding, B. Liu, and L. Zhang. Entity discovery and assignment for opinion mining applications. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1125–1134. ACM, 2009.
- [5] Aidan Finn, Nicholas Kushmerick, and Barry Smyth. Genre classification and domain transfer for information filtering. In Fabio Crestani, Mark Girolami, and Cornelis van Rijsbergen, editors, *Advances in Information Retrieval*, volume 2291 of *Lecture Notes in Computer Science*, pages 349–352. Springer Berlin / Heidelberg, 2002.
- [6] M. Ganapathibhotla and B. Liu. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 241–248. Association for Computational Linguistics, 2008.
- [7] T.R. Gruber et al. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.
- [8] Vasileios Hatzivassiloglou and Kathleen R. McKeown. Predicting the semantic orientation of adjectives. In *Proceedings of the eighth conference on European chap-*

- ter of the Association for Computational Linguistics, EACL '97, pages 174–181, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics.
- [9] Matthew Horridge and Sean Bechhofer. The owl api: A java api for working with owl 2 ontologies.
- [10] Mingqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 168–177, New York, NY, USA, 2004. ACM.
- [11] N. Jindal and B. Liu. Review spam detection. In *Proceedings of the 16th international conference on World Wide Web*, pages 1189–1190. ACM, 2007.
- [12] N. Jindal and B. Liu. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, pages 219–230. ACM, 2008.
- [13] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.
- [14] B. Liu. Sentiment analysis and subjectivity. *Handbook of Natural Language Processing*, pages 978–1420085921, 2010.
- [15] Michael OMahony, Pdraig Cunningham, and Barry Smyth. An assessment of machine learning techniques for review recommendation. In Lorcan Coyle and Jill Freyne, editors, *Artificial Intelligence and Cognitive Science*, volume 6206 of *Lecture Notes in Computer Science*, pages 241–250. Springer Berlin / Heidelberg, 2010.
- [16] I. Peñalver-Martínez, R. Valencia-García, and F. García-Sánchez. Ontology-guided approach to feature-based opinion mining. *Natural Language Processing and Information Systems*, pages 193–200, 2011.
- [17] Ana-Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 339–346, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.
- [18] G. Qiu, B. Liu, J. Bu, and C. Chen. Expanding domain sentiment lexicon through double propagation. In *International Joint Conference on Artificial Intelligence (IJCAI-09)*, 2009.

- [19] Ellen Riloff, Janyce Wiebe, and Theresa Wilson. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4, CONLL '03*, pages 25–32, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.
- [20] Kristina Toutanova and Christopher D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13, EMNLP '00*, pages 63–70, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.
- [21] Dong Wang and Yang Liu. A cross-corpus study of unsupervised subjectivity identification based on calibrated em. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2.011)*, pages 161–167, Portland, Oregon, June 2011. Association for Computational Linguistics.
- [22] G. Wu, M. Harrigan, and P. Cunningham. A characterization of wikipedia content based on motifs in the edit graph. 2011.
- [23] L. Zhang and B. Liu. Identifying noun product features that imply opinions. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 575–580. Association for Computational Linguistics, 2011.
- [24] L. Zhang, B. Liu, S.H. Lim, and E. O'Brien-Strain. Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1462–1470. Association for Computational Linguistics, 2010.
- [25] L. Zhao and C. Li. Ontology based opinion mining for movie reviews. *Knowledge Science, Engineering and Management*, pages 204–214, 2009.
- [26] L. Zhuang, F. Jing, and X.Y. Zhu. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 43–50. ACM, 2006.