

# **A Vector Space Approach for Aspect-Based Sentiment Analysis**

by  
Abdulaziz Alghunaim

B.S., Massachusetts Institute of Technology (2015)

Submitted to the  
Department of Electrical Engineering and Computer Science  
in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology

May 2015

©Massachusetts Institute of Technology 2015. All rights reserved.

The author hereby grants to M.I.T. permission to reproduce and to distribute  
publicly paper and electronic copies of this thesis document in whole and in part  
in any medium now known or hereafter created.

Author .....

Department of Electrical Engineering and Computer Science  
21<sup>st</sup> May, 2015

Certified by .....

Mitra Mohtarami  
Postdoctoral Associate, Co Thesis Supervisor

Certified by .....

James Glass  
Senior Research Scientist, Co Thesis Supervisor

Accepted by .....

Prof. Albert R. Meyer  
Chairman, Masters of Engineering Thesis Committee



# A Vector Space Approach for Aspect-Based Sentiment Analysis

by Abdulaziz ALGHUNAIM

Submitted to the Department of Electrical Engineering and Computer Science

21<sup>st</sup> May, 2015

In Partial Fulfillment of the Requirements for the Degree of Master of Engineering in  
Electrical Engineering and Computer Science

## *Abstract*

Vector representations for language have been shown to be useful in a number of Natural Language Processing (NLP) tasks. In this thesis, we aim to investigate the effectiveness of word vector representations for the research problem of Aspect-Based Sentiment Analysis (ABSA), which attempts to capture both semantic and sentiment information encoded in user generated content such as product reviews. In particular, we target three ABSA sub-tasks: aspect term extraction, aspect category detection, and aspect sentiment prediction. We investigate the effectiveness of vector representations over different text data, and evaluate the quality of domain-dependent vectors. We utilize vector representations to compute various vector-based features and conduct extensive experiments to demonstrate their effectiveness. Using simple vector-based features, we achieve F1 scores of 79.9% for aspect term extraction, 86.7% for category detection, and 72.3% for aspect sentiment prediction.

Co Thesis Supervisor: James Glass  
Title: Senior Research Scientist

Co Thesis Supervisor: Mitra Mohtarami  
Title: Postdoctoral Associate



## *Acknowledgements*

I would like to start by thanking God (the ever most Generous) for enabling me to be at this institution and blessing me with the health and mind to work on this thesis.

There are many people who contributed greatly to my work on this thesis. I owe my gratitude to all of them who have made this thesis possible, and for contributing to my great experience here at MIT.

My deepest gratitude is for my advisers, James Glass and Mitra Mohtarami. James has been a great mentor since I started working with him two years ago. I am thankful for his dedication towards the work of his students and his valuable contributions towards my research interests. From the time I was an undergraduate researcher with Jim, throughout my work on my master's degree, Jim would meet with me on an almost weekly basis to discuss my work. His flexibility and support for my shifting research topics was the only motivation I needed to start exploring a new problem. As I move on in life, I hope that one day I will be as good of a mentor to someone as Jim was to me.

My co-adviser, Mitra, has been working with me since the first day on this thesis topic. Throughout the work on this thesis, Mitra provided me with day-to-day mentorship and guidance. She was very dedicated to the success of this work and she would invest a great deal of her time pointing me to different resources and explaining many foreign concepts to me. Having access to someone with her expertise to discuss design of experiments and ideas behind new features was a blessing. Mitra always made time to talk to me. She pushed me to go beyond what I think I could have achieved. I can simply say, without her, this thesis would not have existed. Thank you, Mitra.

I am grateful to Scott Cyphers for being my go-to for technical questions. Scott helped me with all sorts of technical problems, from setting up servers to debugging my code. He saved me tens of hours. Scott also went through the agony of proofreading my writing. He carefully read and commented my papers, proposals, and this thesis. Importantly, Scott made lab fun for me: he would always have some interesting story to share early in the morning, about his observations about the construction outside his window, some interesting fact about Boston's history, or how MIT policies changed over the past 20 years. I learned many, many things from him.

Many friends have helped me stay sane over the years here at MIT. Their support and motivation was my fuel. I especially thank: Abdullah, Atif, Abubakar, and Muneeza for being there for me.

Finally, none of this would have been possible without the support of my family. Words fail to pay them justice for the love they have blessed me with. Mom, Dad, Nada, Sarah,

Ziyad, Faisal, Norah, and Dahoom were the main reason I was able to come to MIT. Their support is far more impactful than they probably realize.

# Contents

<b>Abstract</b>	<b>3</b>
<b>Acknowledgements</b>	<b>5</b>
<b>Contents</b>	<b>7</b>
<b>List of Figures</b>	<b>11</b>
<b>List of Tables</b>	<b>13</b>
<b>1 Introduction</b>	<b>17</b>
1.1 Motivation . . . . .	17
1.2 Goal . . . . .	19
1.3 Task Description . . . . .	19
1.3.1 Aspect Term Extraction . . . . .	20
1.3.2 Aspect Sentiment Prediction . . . . .	20
1.3.3 Aspect Category Detection . . . . .	20
1.4 Contributions . . . . .	21
1.5 Thesis Outline . . . . .	21
<b>2 Related Work</b>	<b>23</b>
2.1 Aspect Based Sentiment Analysis . . . . .	23
2.2 Vector Space Methods . . . . .	25
<b>3 Experimental Methods</b>	<b>27</b>
3.1 System Architecture . . . . .	27
3.2 A Vector Space Approach for Aspect-Based Sentiment Analysis . . . . .	28
3.2.1 Aspect Term Extraction . . . . .	29
3.2.1.1 Conditional Random Fields . . . . .	30
3.2.1.2 Hidden Markov Support Vector Machines . . . . .	31
3.2.2 Aspect Sentiment Prediction . . . . .	32
3.2.2.1 One-vs-all SVM . . . . .	33
3.2.3 Aspect Category Detection . . . . .	35
3.2.3.1 Multilabel SVM . . . . .	37
3.3 Extra Features . . . . .	37

---

3.3.1	Aspect Term Extraction . . . . .	37
3.3.2	Aspect Sentiment Prediction . . . . .	38
3.3.3	Aspect Category Detection . . . . .	39
<b>4</b>	<b>Evaluation and Results</b>	<b>41</b>
4.1	Task and Data . . . . .	41
4.1.1	Restaurant Review Dataset . . . . .	41
4.1.2	Word2Vec Datasets . . . . .	42
4.2	Experimental Platform . . . . .	43
4.2.1	Readers . . . . .	44
4.2.2	Annotators . . . . .	44
4.2.3	Feature Extractors . . . . .	44
4.2.4	Classifiers . . . . .	45
	Aspect Term Extraction (Task 1) . . . . .	45
	Aspect Sentiment Prediction (Task 2) . . . . .	46
	Aspect Category Detection (Task 3) . . . . .	47
4.3	Task 1: Aspect Term Extraction . . . . .	48
4.3.1	Evaluation Metrics . . . . .	48
4.3.2	Aspect Term Extraction Results . . . . .	49
4.4	Tasks 2: Aspect Sentiment Prediction . . . . .	51
4.4.1	Evaluation Metric . . . . .	51
4.4.2	Aspect Sentiment Prediction Results . . . . .	51
4.5	Task 3: Aspect Category Detection . . . . .	53
4.5.1	Evaluation Metric . . . . .	53
4.5.2	Aspect Category Detection Results . . . . .	53
4.6	Extra Features Results . . . . .	55
4.6.1	Aspect Term Extraction-Extra Features . . . . .	55
4.6.2	Aspect Sentiment Prediction-Extra Features . . . . .	57
4.6.3	Aspect Category Detection-Extra Features . . . . .	59
4.7	Discussion of Results . . . . .	59
<b>5</b>	<b>System Demonstration</b>	<b>61</b>
<b>6</b>	<b>Conclusion</b>	<b>63</b>
6.1	Summary . . . . .	63
6.2	Future Work . . . . .	64
6.2.1	Extending the Experiments . . . . .	64
6.2.1.1	Text-based Features . . . . .	64
6.2.1.2	Extra Data Sources . . . . .	65
6.2.2	Extending the Research Problem . . . . .	66
6.2.2.1	Category Sentiment Prediction . . . . .	66
6.2.2.2	Aspect Sentiment Summarization . . . . .	67
<b>A</b>	<b>Extra Data Sources</b>	<b>69</b>
A.1	Dining In Doha . . . . .	69
	About the Restaurant . . . . .	69

Restaurant Details . . . . .	69
Our rating . . . . .	70
User Rating . . . . .	70
Reviews/Comments . . . . .	70
A.1.1 Data Collection . . . . .	70
A.2 Labeling Reviews on Amazon Mechanical Turk . . . . .	71
<b>Bibliography</b>	<b>75</b>



# List of Figures

1.1	Applying our model to reviews can give us a deeper understanding about users' sentiment towards the entity. In this example, we see the different aspects of the restaurant highlighted with the sentiment attached with them . . . . .	18
1.2	This chart shows the distribution of confidence about what categories the review is speaking about and their associated sentiment. This chart shows that <i>Food</i> and <i>Price</i> have negative sentiments while <i>Ambiance</i> has a <i>positive</i> sentiment. . . . .	19
1.3	Illustration of the tasks we are performing. We start by extracting the aspect terms from a review (task 1). Then we are interested in predicting the sentiment of the aspect terms (task 2). Also, we are interested in identifying the aspect categories discussed in the review from a set of predefined categories (task 3). . . . .	21
3.1	Illustration of development framework using UIMA and dkpro-tc. The sentences will feed to UIMA to compute the features of interest and we will use the features to start an iterative process of classification and picking the best set of features until we have the set of features that produces the best classification. . . . .	28
3.2	The dependency tree for the review "Certainly not the best sushi in city, however, it is always fresh, and the place is very clean, sterile." generated using the Stanford Parser. We use this dependency tree to find the dependency words of the aspects. Aspect terms in this tree are highlighted by their sentiment (e.g. "place" and "sushi" have positive and conflict sentiments, respectively). Each aspect term has a set of dependency words, and they are defined as the words connected to the aspect term in the dependency tree. . . . .	32
3.3	On the left hand side of the figure we see how ADV is computed using a vector model that is trained on the entire review dataset. In RV, we train four different vector models each trained on one rating level (The rating 3 is ignored, because it mostly contains neutral reviews, so it includes both positive and negative words). Then, we compute the average dependency vector four times, once with each of the models. . . . .	33
3.4	Set of positive and negative seed words. The positive seed words were found by retrieving the top 20 nearest neighbor word vectors to the vector of the word "excellent." Similarly, the negative seed words were retrieved with respect to the vector of the word "poor." . . . . .	34

---

3.5	This figure shows the projection of category representative vectors on two dimensions. Each cluster represents the set of top 20 vectors that represent that corresponding category. Table 3.1 lists all the representative words. . . . .	36
4.1	The plot shows the distribution of number of aspects in a review in both the training and test datasets. In total, the training dataset has 3,693 aspects and the test dataset has 1,025 aspect terms. . . . .	43
4.2	The figure illustrates the work-flow of the dkpro-tc application we developed. In our pipeline, the cycle starts at a Reader that process the input data, then passes that to the Annotators in order to perform pre-processing on the text. The Annotators pass the output to a Feature Extractor that computes a feature vector from the annotated text. Finally the feature vectors are passed to the classifier. . . . .	45
4.3	This plot shows the recall, precision, and F1 values for different values of the C parameter for the SVM-HMM aspect term extraction model. . . . .	46
4.4	This plot shows the average accuracy for 5-fold cross validation for every C value. In all the experiments we used the Average Dependency Vector (ADV), and the Rating Vectors (RV) as the features. We set $C = 0.1$ . . . . .	47
4.5	Results for 5-fold cross validation for tuning C in Category Detection subtask. We are plotting the average of the 5 experiments using that specific value of $C$ . We used the number of tokens, category similarities, and normalized average vector as the features. There is a wide range for possible $C$ values, we set $C = 0.25$ . . . . .	48
4.6	This figure shows the aspect term extraction model applied to the phrase “BEST spicy tuna roll, great asian salad.” We can see that our prediction model was able to detect both multi-word aspects, but for one of them it did not label all the words correctly. . . . .	51
4.7	This figure shows the aspect polarity prediction model output on reviews from the test data set. In the second example we can see that our model is biased towards positive sentiments. . . . .	53
4.8	This figure lists two example reviews from the test data set. For the first one, the model predicts the correct category of <i>Service</i> by seeing the word “staff” in the review. In the second one, the model mistakenly predicts the category <i>Ambiance</i> because of the word “feel” in the review. . . . .	55
5.1	Screenshot of the online system demo. . . . .	62
A.1	This figure is a screenshot of what a restaurant page on <a href="http://diningindoha.com">diningindoha.com</a> looks like. . . . .	72
A.2	Screenshot of the Amazon Mechanical Turk labeling task. The worker highlights the phrase and indicates whether it is positive or negative. Unlabeled text is assumed to be neutral. . . . .	73

# List of Tables

1.1	A sample restaurant review that illustrates the subtasks of Aspect-Based Sentiment Analysis. . . . .	20
3.1	This table lists the category representative words for each one of the categories. Figure 3.5 projects the vectors of those words on a two dimensional space. . . . .	36
3.2	List of most common negation words. . . . .	39
4.1	Category distributions over the dataset. . . . .	42
4.2	(a) Number of reviews that contain multiple categories in each subset. (b) Percentage of reviews that contain multiple categories in each subset. . . . .	42
4.3	This table shows the results of using three different algorithms to train the CRF. Each CRF model was trained on the entire training set using 5-fold cross validation and using two types of features, the POS tag and the word vector representation. From this table, we see that lbfgs algorithm has the best performance. . . . .	46
4.4	Aspect-Level Evaluation results for the aspect term extraction task. . . .	50
4.5	Word-Level Evaluation results for the aspect term extraction task. . . .	50
4.6	Results for the aspect sentiment prediction task. . . . .	52
4.7	Results for the aspect category detection task. . . . .	54
4.8	Results from testing extra features on the aspect term extraction subtask. . .	56
4.9	Results for Aspect Sentiment Prediction subtask using extra features. . .	58
4.10	Results for Aspect Category Detection subtask using extra features. . .	59
A.1	Summary of the data collected from <a href="#">diningindoha.com</a> on February 13, 2014 . . . . .	71



*Dedicated To My Family*



# Chapter 1

## Introduction

In this thesis, we investigate Aspect-Based Sentiment Analysis (ABSA), and explore the effectiveness of using vector-based features to tackle this problem. In this Chapter, we will discuss the motivation for this research, our goals, problem definition, and, finally, our contributions to this topic.

### 1.1 Motivation

*Sentiment analysis*, or *opinion mining*, deals with computational analysis of people's opinions, sentiments, attitudes and emotions towards target entities such as products, organizations, individuals, topics and their attributes (Liu, 2012). The majority of early approaches to this research problem (Baccianella et al., 2009, Pang and Lee, 2005, Pang et al., 2002) attempted to detect the overall sentiment of a sentence, paragraph, or text span regardless of the entities (e.g., *restaurants*) and their aspects (e.g., *food, service*) expressed in context. However, considering only overall sentiments (like the total star ratings of a restaurant, as shown in Figure 1.1) fails to capture the sentiments over the aspects on which an entity can be reviewed (Lu et al., 2011).

In this thesis, we are interested in a more granular approach to analyzing the sentiments captured in user generated restaurant reviews. One of the most exciting applications that motivate us for this research is more effective processing of the increasing amounts of user-generated content on the web. Today more and more people are leaving comments and reviews online in amounts much larger than our ability to read. Every restaurant or product has hundreds, if not thousands, of opinions written about it. Many platforms, such as Amazon<sup>1</sup> and Yelp<sup>2</sup>, try to develop better ways to display opinions to users.

---

<sup>1</sup>website: [amazon.com](http://amazon.com)

<sup>2</sup>website: [yelp.com](http://yelp.com)

One of the most popular techniques is summarizing the information for an entity by looking for similar phrases with high frequency over the reviews with high ratings.

The limitation of these methods is the loss of important information by summarization techniques. Instead, when we apply ABSA on these large sets of reviews, we can easily build representative models for aspects of the entity and associated sentiments.



FIGURE 1.1: Applying our model to reviews can give us a deeper understanding about users' sentiment towards the entity. In this example, we see the different aspects of the restaurant highlighted with the sentiment attached with them

ABSA can automatically extract the aspects in the review and define what the reviewer thought about the aspects' sentiments, as shown in Figure 1.1. Having this knowledge allows us to easily answer questions such as “What is the best dish at restaurant X?” or “Why do people not like the burger at Y?”.

Furthermore, we aggregate specific aspect level information from all the reviews to build an understanding of how people perceive a particular entity. In Figure 1.2 we show an example where we see the associated sentiment for different categories of a restaurant. As Figure 1.2 shows, while the user has negative sentiments towards food and price, they have positive sentiments towards service and ambiance.

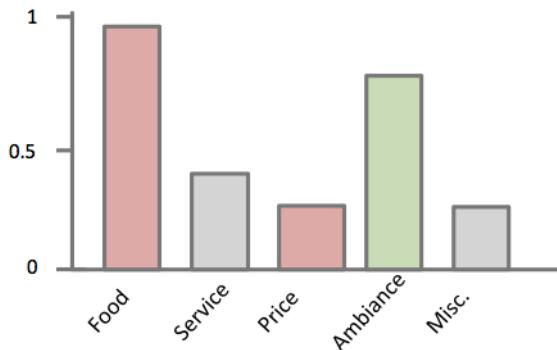


FIGURE 1.2: This chart shows the distribution of confidence about what categories the review is speaking about and their associated sentiment. This chart shows that *Food* and *Price* have negative sentiments while *Ambiance* has a *positive* sentiment.

In conclusion, applying ABSA techniques to the world of online opinion spaces would be extremely beneficial in terms of extracting meaningful information from those large sets of data. It would allow the user to quickly understand the aggregate sentiment about a specific entity and it would also be very helpful to entities to understand the online perception about entities and the drivers behind them.

## 1.2 Goal

Natural Language representation in vector spaces has been successfully used in many NLP tasks. Previous research has employed vector representations to present the *syntactic* and *semantic* information in textual contents. In this thesis, we investigate the effectiveness of vector space representations for *Aspect-Based Sentiment Analysis*, in which we aim to capture both *semantic* and *sentiment* information encoded in user generated content, such as restaurant reviews.

## 1.3 Task Description

In this section, we will describe *Aspect-Based Sentiment Analysis*. ABSA is a new direction in sentiment analysis research. The goal of ABSA is to identify the aspects (or semantic labels) of given target entities and the sentiment expressed towards each aspect. For this purpose, three sub-tasks need to be addressed: (1) *aspect term extraction*, (2) *aspect category detection*, and (3) *aspect sentiment prediction*. We describe those tasks in the following subsections.

---

*Our agreed favorite is the **orecchiette with sausage and chicken** and usually the **waiters** are kind enough to split the **dish** in half so you get to sample both **meats**. But, the **music** which is sometimes a little too heavy for my taste.*

---

TABLE 1.1: A sample restaurant review that illustrates the subtasks of Aspect-Based Sentiment Analysis.

### 1.3.1 Aspect Term Extraction

The objective of this task is to identify the aspect terms (or semantic labels) appearing in a given text about a target entity.

Given a set of sentences that target a specific pre-identified entity (e.g. a restaurant review), we need to identify all the aspect terms in that set and return a list of distinct aspect terms. For instance, in the review in Table 1.1, the aspects are “orecchiette with sausage and chicken”, “waiters”, “dish”, “meats” and “music”, and the target entity is “restaurant”. In addition, multi-word aspect terms are treated as a single aspect, like “orecchiette with sausage and chicken” in the example is all considered one aspect.

### 1.3.2 Aspect Sentiment Prediction

The objective of this task is to identify the sentiment of aspect terms as *positive*, *negative*, *neutral*, or *conflict* (i.e., both positive and negative) for a given set of aspect terms in a text.

Given a set of aspect terms for a specific entity, we need to determine the sentiment assigned to each unique aspect. Each aspect can be assigned one of the following sentiments: positive, negative, neutral, or conflict; where in conflict the aspects have both positive and negative sentiments. For example, in the review in Table 1.1, the aspects “orecchiette with sausage and chicken” and “waiters” are positive, while “music” is negative, and “dish” and “meats” are neutral.

### 1.3.3 Aspect Category Detection

The objective of aspect category detection is to identify (latent) aspect categories available in a given text. Aspect categories are coarser than aspect terms, and they do not necessarily occur as terms in the text. For example, the review in Table 1.1 contains the latent aspect categories “food”, “service”, and “ambiance”. Aspect categories are often considered as predefined categories (e.g. “price”, “food”) with respect to the target entities.

Figure 1.3 highlights a high-level overview of the subtasks we need to perform for ABSA. It starts with collecting the data we are interested in analyzing. Next, Aspect Term Extraction is performed. Then, once we have the aspects, we can predict the sentiment of the aspects as well as detecting the categories discussed in the review.

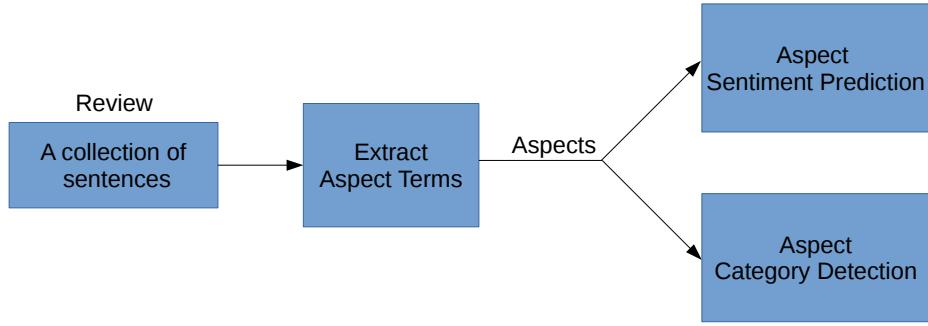


FIGURE 1.3: Illustration of the tasks we are performing. We start by extracting the aspect terms from a review (task 1). Then we are interested in predicting the sentiment of the aspect terms (task 2). Also, we are interested in identifying the aspect categories discussed in the review from a set of predefined categories (task 3).

## 1.4 Contributions

In this thesis, we employ word vector representations to compute vector-based features to tackle the problem of *Aspect-Based Sentiment Analysis*. We successfully introduced several effective vector-based features and showed their utility in addressing aspect term extraction, aspect category detection, and aspect sentiment prediction on a publicly available corpus of restaurant reviews. Our vector space approach using these features performed well compared to the baselines. We achieve F1 scores of 79.9% for aspect term extraction compared to a baseline of 47.1%. In Aspect Category Detection, we get an F1 score of 86.7% compared to a baseline of 65.6%. Finally, for Aspect Sentiment Prediction we achieve a 72.3% compared to a 64.2% baseline.

## 1.5 Thesis Outline

In this thesis, we first review of prior research in sentiment analysis, and ABSA , and also discuss the concept behind word vector representations. Chapter 3 describes the general

architecture of our ABSA system, and the methods and algorithms we implemented to design our experiments. It describes the feature specifications for each of the tasks, and describes the algorithms used in each task. In Chapter 4, we evaluate our vector space method on restaurant reviews, and discuss the results and best performance settings. Chapter 5 describes an online application built to demonstrate the output of our ABSA platform. Finally, we conclude, and discuss future work in Chapters 6 and 7.

# Chapter 2

## Related Work

Since early 2000, sentiment analysis has become the most active research area in natural language processing. This growth is mainly due to the social media revolution that generates large volumes of opinionated data. Sentiment analysis has become a focus of social media research. From early 2000, it has found its way into a number of other fields, including management sciences, political science, social and economic sciences (Liu, 2012).

Nowadays, interest in sentiment analysis spans many domains. Since the concept of “opinion” is critical to many activities, businesses and other entities are interested in knowing what those opinions are. Researchers have applied sentiment analysis in many real-life domains, such as predicting sales performance (Liu et al., 2007), linking Twitter sentiments with public opinion polls (O’Connor et al., 2010), predicting box-office revenues (Doshi, 2010), predicting the stock market (Bollen et al., 2011), studying trading strategies (Zhang and Skiena, 2010), and studying the relationship between the NFL betting line and public opinions (Hong and Skiena, 2010).

Our research touches on ideas in both sentiment analysis, and continuous vector representation of words. In this section we will present related work in these two areas.

### 2.1 Aspect Based Sentiment Analysis

Sentiment analysis started with an interest in knowing the sentimental polarity of a document. Sentiment polarity indicates whether a given text holds *positive*, *negative*, or *neutral* sentiment. One of the early sentiment analysis works applied a number of machine learning methods to determine the polarity of movie reviews (Pang et al., 2002). Their work was motivated by text categorization; they were interested in finding novel

methods to categorize unstructured text, and sentiment was one aspect for categorizing documents.

Another early work looked at reviews in general, and was interested in understanding if a review recommended a product/service, or did not recommend it (thumbs up or down as they call it). They simply calculated the mutual information between the review and the word “excellent” minus the mutual information between the review and the word “poor.” A review was recommended if the calculated quantity is positive (the review had more mutual information with “excellent” compared to “poor”). The authors applied their algorithm on reviews from many domains including automobiles, banks, movies, and travel (Turney, 2002).

Although work on document-level sentiment analysis is able to generate good levels of accuracy, it misses something critical. It fails to adequately represent the multiple potential dimensions on which an entity can be reviewed, what people liked or did not like. It does not expose the source of the opinion, and, rather, reports just the overall sentiment. For example, although the restaurant review shown in Table 1.1 might express an overall *positive* sentiment, it specifically expresses *positive* sentiment toward the restaurant’s *food* and *service*, as well as *negative* sentiment toward the restaurant’s *ambiance*.

Because of the limitations in document-level sentiment analysis, researchers started investigating finer-grained methods. (Hu and Liu, 2004) built a review summarization system that takes all the reviews of a product and summarizes the features that had a sentiment referring to them and whether it was positive or negative. Also, Popescu and Etzioni (Popescu and Etzioni, 2007) developed an unsupervised system to extract the product features and opinions from online reviews. This research direction introduced the field of Aspect Based Sentiment Analysis (ABSA). The idea behind ABSA is to first perform aspect extraction to identify the aspect terms in the document. Next, the aspect term sentiments are classified. ABSA is critical to understanding the opinions around online user-generated content (Gamon et al., 2005).

Previous works on ABSA (Liu, 2012, Pang and Lee, 2008) attempted to tackle sentiment and semantic labeling using different approaches, such as sequence labeling (Choi and Cardie, 2010, Yang and Cardie, 2013), syntactic patterns (Xu et al., 2013, Zhao et al., 2012, Zhou et al., 2013), and topic models (Lu et al., 2011). While some works first separate the semantic and sentiment information and then label them (Mei et al., 2007, Zhao et al., 2010), other works developed models for joint semantic and sentiment labeling (Jo and Oh, 2011, Lin and He, 2009).

Over the past couple of years a number of ABSA applications were developed in several domains:

- Movie reviews (Thet et al., 2010)
- Customer reviews of electronic products i.e. digital cameras (Hu and Liu, 2004)
- Netbook computers (Brody and Elhadad, 2010)
- Services (Long et al., 2010)
- Restaurants (Brody and Elhadad, 2010, Ganu et al., 2009b)

In this thesis, we attempt to investigate the ABSA problem as three subtasks (i.e., aspect term extraction, aspect sentiment prediction, and aspect category detection) using a publicly available corpus of restaurant reviews (Pontiki et al., 2014).

## 2.2 Vector Space Methods

The research in this thesis investigates the impact of word representation techniques for ABSA. In particular, we are interested in employing recursive neural networks (RNNs) to generate vector-based features over word representations.

Vector representations for words and phrases have been found to be useful for many NLP tasks (Al-Rfou et al., 2013, Bansal et al., 2014, Bowman et al., 2014, Boyd-Graber et al., 2012, Chen and Rudnicky, 2014, Guo et al., 2014, Iyyer et al., 2014, Levy and Goldberg, 2014). Collobert and Weston showed that one particular neural network model can obtain state of the art results on a number of tasks, such as named entity recognition and parts of speech tagging (Collobert and Weston, 2008). Those results were obtained by representing words as continuous vectors compared to the more common bag-of-words discrete vector representation.

In 1957, Firth introduced the idea of representing a word as a function of its neighbors to produce word vectors that can capture both semantics and co-occurrence statistics. This idea proved very powerful recently in NLP, for example, the idea of neural language models (which outperform n-gram models) is to jointly learn word vectors and use them to predict how likely a word occurs given its context (Bengio et al., 2003). It was also shown that the combination of word vectors and neural networks can be used to classify words into different categories, such as parts of speech tags, or named entity tags (Collobert et al., 2011).

Researchers also developed compositional methods to combine word vectors into phrase vectors. Those vectors are then used with a Recursive Neural Tensor Network (RNTN) and the Stanford Sentiment Treebank to predict fine-grained sentiment labels with high accuracy (Socher et al., 2013). The fine-grained sentiment labels allow us to know the sentiment label of every phrase in the sentence, but it still does not achieve the objective of ABSA. Basically, we still do not have aspects of the target with associated sentiment. Instead we have the sentiment of all phrases in the sentence.

To extend on Socher’s work, another group tried to perform ABSA using the Stanford Sentiment Treebank. The aspect extraction was using a simple rule-based system. After extracting the aspect term they built a sentiment tree and traversed it from the aspect node to the root node, returning the first non-neutral sentiment they found. If all sentiments up to the root are neutral, then the aspect is reported as neutral (Pontiki et al., 2014). This work did not prove to be very accurate as it obtained an F-score of 0.48 with an accuracy of 0.62 for aspect polarity.

In this thesis we investigated the impact of word representation techniques for ABSA. In particular, we aimed to employ vector-based features using word representations to capture both semantic and sentiment information.

# Chapter 3

## Experimental Methods

This Chapter outlines the methods used in this thesis to implement the Aspect-Based Sentiment Analysis (ABSA) system. It illustrates the general structure for the approach we take to tackle each of the sub-tasks. Vector space methods are discussed in detail, followed by a description for the models used in each of the sub-tasks.

### 3.1 System Architecture

The general structure of our approach is shown in Figure 3.1, As shown in the figure, we start by processing the user generated content (e.g. restaurant reviews) using the Unstructured Information Management Architecture (UIMA) framework and dkproto framework (Daxenberger et al., 2014). UIMA is a framework architecture built to support analysis of unstructured data (Ogren and Bethard, 2009). It defines interfaces and guidelines for designing analysis components as well as easy access to libraries of recyclable components. Each component is called an Analysis Engine (AE), and takes input text and performs a specific analysis task on it. For example one AE can perform tokenization and pass the output to another AE to perform parts of speech tagging, and so on. After computing all the features of interest we generate vector-based features vectors and pass them to a classifier, as shown in Figure 3.1. We implement a feature engineering cycle to understand each feature's effect on the classification results and report the best features alongside highest performing classification.

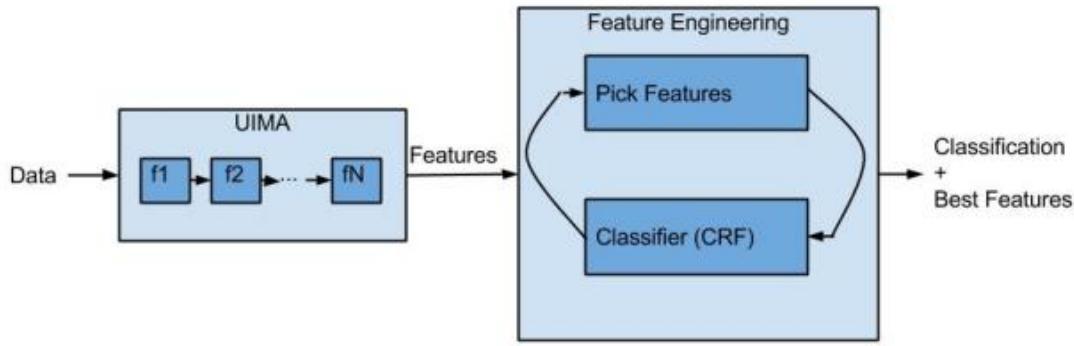


FIGURE 3.1: Illustration of development framework using UIMA and dkpro-tc. The sentences will feed to UIMA to compute the features of interest and we will use the features to start an iterative process of classification and picking the best set of features until we have the set of features that produces the best classification.

## 3.2 A Vector Space Approach for Aspect-Based Sentiment Analysis

Distributed vector representations, described by Schütze (Schütze, 1992a,b), associate similar vectors with similar words and phrases. These vectors provide useful information for the learning algorithms to achieve better performance in NLP tasks (Mikolov et al., 2013c). Most approaches to computing vector representations use the observation that similar words appear in similar contexts (Firth, 1957, Mikolov, 2012, Sahlgren, 2006, Socher, 2014).

To compute the vector representations of words, we use the skip-gram model of **Word2Vec** (Mikolov, 2014, Mikolov et al., 2013a,b,d). The Skip-gram model aims to find word representations that are useful for predicting the surrounding words in a sentence or document (Mikolov et al., 2013b). The model needs a large amount of unstructured text data for training the word vector representations.

When training the skip-gram model we use the GoogleNews dataset (Mikolov, 2014) that contains 3 million unique words and about 100 billion tokens. To account for the effect of domain information on the quality of word representations, we also use a dataset of restaurant reviews<sup>1</sup> from Yelp that contains 131,778 unique words and about 200 million tokens. We trained 300-dimensional word vectors from these combined data.

We propose to utilize word vector representations to compute vector-based features for the three sub-tasks of ABSA. We employ these features in a supervised learning setting to address the tasks. Our data (reviews) are first analyzed by the Stanford

<sup>1</sup>This dataset is available on: [http://www.yelp.com/dataset\\_challenge](http://www.yelp.com/dataset_challenge).

tokenizer (Manning et al., 2010), POS-tagger (Toutanova et al., 2003) and dependency-tree extractor (de Marneffe and Manning, 2008). Then, the pre-processed data and word representations are used to compute task-specific features as explained in the following subsections.

### 3.2.1 Aspect Term Extraction

The objective of this sub-task is to extract aspect terms from reviews with respect to a target entity (e.g, restaurant) as explained in Chapter 1. This task can be considered as part of Semantic Role Labeling (SRL). Previous research has shown that Conditional Random Fields (CRFs) (Lafferty et al., 2001) and sequence tagging with Structural Support Vector Machines (SVM-HMM) (Altun et al., 2003a) are effective for the SRL task (Cohn and Blunsom, 2005). As such, we employ CRFsuite (Okazaki, 2007) and SVM-HMM (Altun et al., 2003a) with word vector representations as features to label the token sequence with respect to two possible tags: “Aspect” and “Not-Aspect”, where an aspect can be multi-word.

We can formulate the Aspect Term Extraction subtask as a single-label sequence labeling task where the input is the sequence of words

$$S = (w_1, w_2, \dots, w_M)$$

For each word  $w_i$  in the sequence, we compute the feature vector  $\vec{x}_i$  resulting the following feature matrix

$$X = (\vec{x}_1, \vec{x}_2, \dots, \vec{x}_M), \text{s.t. } \vec{x}_i \text{ is a feature vector}$$

and the output is the label sequence

$$y = (y_1, y_2, \dots, y_M)$$

where  $y_i \in \{\text{Aspect}, \text{Not Aspect}\}$  for all  $i$ , such that each word is labeled either as an Aspect or Not Aspect.

To the best of our knowledge, this is the first attempt to solve aspect term extraction using CRFsuite or SVM-HMM with vector representations as features. In addition to the vector-based features, we employ POS-tags information as an additional feature. This is mainly because “nouns” are strong indicators for aspects (Blinov and Kotelnikov, 2014,

Pontiki et al., 2014). However, as we will discuss in Chapter 4, this feature is more effective for the single term aspects.

Given that the Aspect Term Extraction subtask is formulated as a one-label sequence tagging problem, we will be exploring two models that solve the sequence tagging problem: the Conditional Random Fields and the Hidden Markov Support Vector Machines.

### 3.2.1.1 Conditional Random Fields

Statistical models are able to learn patterns from data sets. In this subsection, we give a quick overview of the statistical model we use in this study, Conditional Random Field (CRF). Given a data set, the CRF model is able to learn from the feature patterns, in the data set and utilize that to do tasks on unseen data. Given a set of tagged data, we would like our model to learn how to correctly tag an unlabeled sentence it has not seen before. There are many methods for developing such models, mainly generative models and classification models. Conditional Random Fields (CRF) bring together the best of generative and classification models by combining key aspects of both. (Sha and Pereira, 2003)

The objective of the CRF is as follows:

given the word feature sequence

$$X = (\vec{x}_1, \vec{x}_2, \dots, \vec{x}_N)$$

find label sequence  $s = (s_1, s_2, \dots, s_N)$ , which is a classification for each of the segments  $1, 2, \dots, M$ . The CRF models the conditional probability  $p(s|x)$  and as such finds  $s$  that maximizes

$$p(s|x) = \frac{1}{Z_\lambda(x)} \exp \sum_{i=1}^{N+1} \lambda f(s_{i-1}, s_i, x, i) \quad (3.1)$$

Where  $\frac{1}{Z_\lambda(x)}$  is a normalization factor,  $\lambda$  is a weight-vector, and  $N$  is the number of words features we have in our sequence of interest (note that  $N + 1$  denotes the end of sequence mark).

The CRF solves two problems at the same time, the  $s$  vector contains both the segmentation of the input vector  $x$  and each segment's corresponding label. Thus the CRF first segments the sentences, then labels it. The CRF is able to do so by using feature functions  $f(s_{i-1}, s_i, x, i)$ . Feature functions that can be used in the CRF can be either

binary feature functions (examples of feature functions can be: is word  $w$  in my dictionary? Is word  $w$  followed by a pronoun? Is  $w$  the beginning of a sentence?) or real valued feature functions. The choice of proper feature functions is dependent on the task at hand.

Given a set of training data  $\{(x, s)\}$ , the CRF estimates  $\lambda$  that maximizes the conditional probability. This  $p(s|x)$  model is used later on to segment and label unseen sequences.

### 3.2.1.2 Hidden Markov Support Vector Machines

While traditional Hidden Markov Models are a very powerful successful model used in a range of applications, it suffers from a number of limitations. Mainly, (i) they are typically trained in a non-discriminative manner, (ii) the conditional independence assumptions are often too restrictive, and (iii) they are based on explicit feature representations and lack the power of kernel-based methods.

Altun, 2003 introduced a novel method for learning label sequences by combining Hidden Markov Models (HMM) with Support Vector Machines (SVMs). This formulation addresses all the previous limitations while retaining the Markov chain dependency structure between labels and an efficient dynamic programming formulation. (Altun et al., 2003b). The objective of the SVM-HMM is as follows:

Given an observed input sequence  $X = (\vec{x}_1 \dots \vec{x}_l)$  of feature vectors  $\vec{x}_1 \dots \vec{x}_l$ , it predicts a tag sequence  $y = (y_1 \dots y_l)$  according to the following linear discriminant function:

$$y^* = \underset{y}{\operatorname{argmax}} \left\{ \sum_{i=1}^l \left[ \sum_{j=1}^k (\vec{x}_i \bullet \vec{w}_{y_{i-j} \dots y_i}) + \vec{\phi}_{trans}(y_{i-j}, \dots, y_i) \bullet \vec{w}_{trans} \right] \right\} \quad (3.2)$$

The SVM-HMM learns one emission weight vector  $\vec{w}_{y_{i-k} \dots y_i}$  for each different  $k$ th-order tag sequence  $y_{i-k} \dots y_i$  (in this work we use a first-order model  $k = 1$ ) and one transition weight vector  $\vec{w}_{trans}$  for the transition weights between adjacent tags.  $\vec{\phi}_{trans}(y_{i-j}, \dots, y_i)$  is an indicator vector that has exactly one entry set to 1 that corresponds to the sequence  $y_{i-j}, \dots, y_i$ . Note that in contrast to a conventional HMM, the observations  $\vec{x}_1 \dots \vec{x}_l$  can naturally be expressed as feature vectors, not just as atomic tokens. See (Joachims et al., 2009) for more details on SVM-HMM, including some part-of-speech tagging experiments.

### 3.2.2 Aspect Sentiment Prediction

The objective of this task is to predict the sentiments for a given set of aspects in a sentence as *positive*, *negative*, *neutral* and *conflict* (i.e., both positive and negative) as explained in Chapter 1. Given that this task is formulated as a single-class classification problem we will be using the SVM method to perform this classification.

For this task, we apply a one-vs-all SVM, as explained in subsection 3.2.2.1), and the following vector-based features for a given aspect:

- **Average Dependency Vector (ADV)** is obtained by averaging the vector representations of the *dependency words* (DW) for the aspect. We define the dependency words for an aspect as the words that modify or are modified by the aspect in the dependency tree of the input sentence. Figure 3.2 shows the dependency tree of an example review. The aspect terms in the tree are highlighted, and the dependency words are all the nodes directly connected to an aspect term. For the example in Figure 3.2, the dependency words for “sushi” are {fresh, best, the, not} and for “place” they are {clean, the}

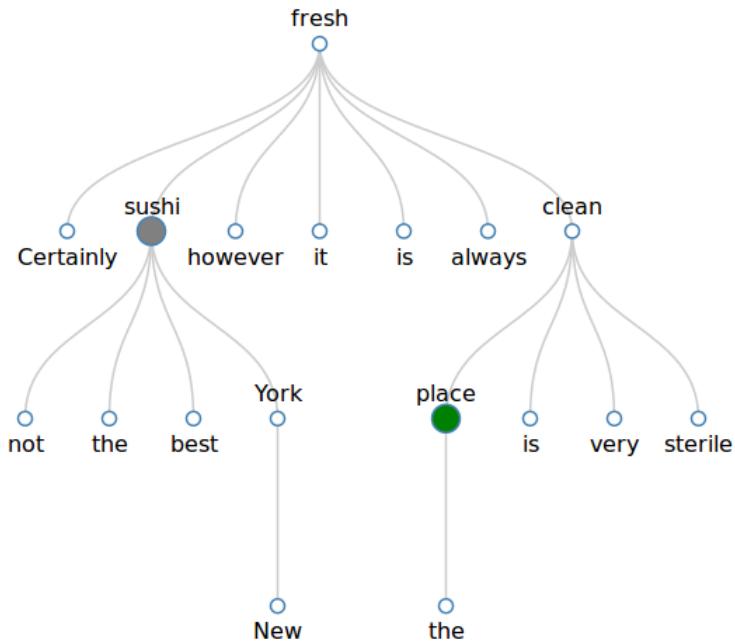


FIGURE 3.2: The dependency tree for the review “Certainly not the best sushi in city, however, it is always fresh, and the place is very clean, sterile.” generated using the Stanford Parser. We use this dependency tree to find the dependency words of the aspects. Aspect terms in this tree are highlighted by their sentiment (e.g. “place” and “sushi” have positive and conflict sentiments, respectively). Each aspect term has a set of dependency words, and they are defined as the words connected to the aspect term in the dependency tree.

- **Rating Vectors (RV)** are the same as ADV features except that they are computed using the vector representations trained on different subsets of our data. We have five subsets, each subset contains only reviews with a specific review rating. Ratings range from 1 (strong negative) to 5 (strong positive). Previous research showed the impact of the word ( $w$ ) distribution over different ratings ( $r$ ) to compute the sentiment of the word (i.e.,  $P(r|w)$ ) (de Marneffe et al., 2010) and construct opinion lexicon (Amiri and Chua, 2012). Using this feature, we can investigate the distribution of words and their vector representations in different ratings. Figure 3.3 illustrates the difference between ADV and RV.

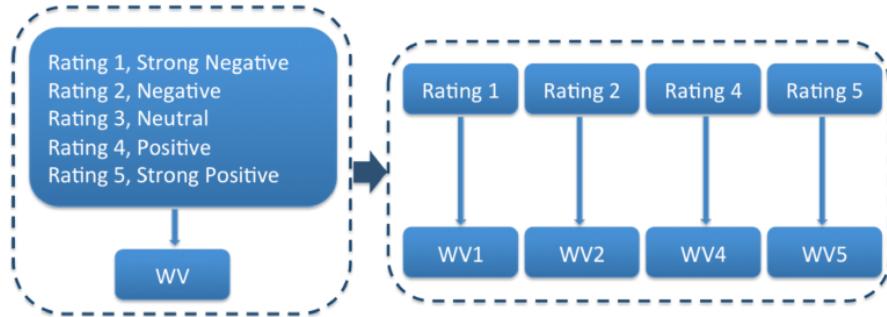


FIGURE 3.3: On the left hand side of the figure we see how ADV is computed using a vector model that is trained on the entire review dataset. In RV, we train four different vector models each trained on one rating level (The rating 3 is ignored, because it mostly contains neutral reviews, so it includes both positive and negative words). Then, we compute the average dependency vector four times, once with each of the models.

- **Positive/Negative Similarities (PNS)** are obtained by computing the highest cosine similarity between DW vectors and the vectors of a set of positive/negative sentiment words. The sentiment words are automatically computed by selecting the top 20 nearest neighbor word vectors to the vectors of the word “excellent” for positive and “poor” for negative seeds, as shown in Figure 3.4.

### 3.2.2.1 One-vs-all SVM

The Aspect Sentiment Prediction subtask is formulated as a standard classification problem at the aspect level. For each aspect in a review we compute a feature vector,

$$X = (\vec{x}_1, \dots, \vec{x}_i, \dots, \vec{x}_N),$$

where  $\vec{x}_i$  corresponds to the feature vector for the  $i$ -th aspect in the review. For each feature vector  $\vec{x}_i$ , the output is a class assignment  $c_i \in \{Positive, Negative, Neutral, Conflict\}$  that corresponds to the  $i$ -th aspect in the review.

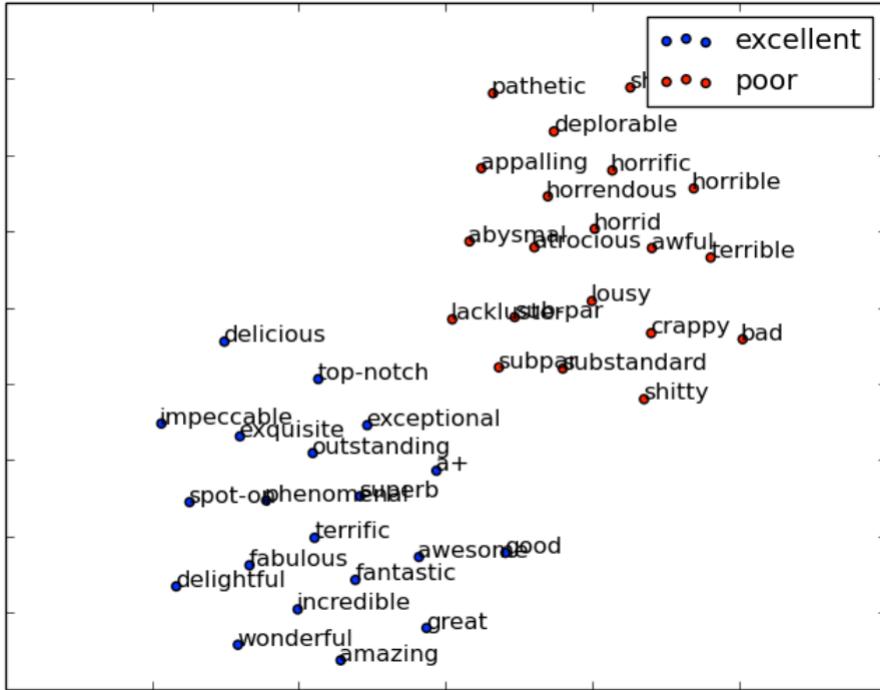


FIGURE 3.4: Set of positive and negative seed words. The positive seed words were found by retrieving the top 20 nearest neighbor word vectors to the vector of the word “excellent.” Similarly, the negative seed words were retrieved with respect to the vector of the word “poor.”

In order to apply a traditional SVM to a multi-class problem (*Positive*, *Negative*, *Neutral*, *Conflict*), we use a one-vs-all SVM algorithm. The algorithm works by constructing  $k$  SVM models where  $k$  is the number of classes ( $k = 4$  in our case). The  $i$ -th SVM model is trained by splitting the training set to two classes. The first class contains data points with the  $i$ -th class label, and the second class contains all other data points (hence the name one-vs-all).

Each one of the SVM models solves the following SVM optimization problem

$$\underset{w^i, b^i, \xi^i}{\text{minimize}} \quad \frac{1}{2} (w^i)^T w^i + C \sum_{j=1}^l \xi_j^i$$

subject to

$$\begin{aligned} (w^i)^T \phi(\vec{x}_j) + b^i &\geq 1 - \xi_j^i, \text{ if } y_j = i, \\ (w^i)^T \phi(\vec{x}_j) + b^i &\leq -1 + \xi_j^i, \text{ if } y_j \neq i, \\ \xi_j^i &\geq 0, j = 1, \dots, l \end{aligned}$$

where the  $\phi$  function maps the feature vector  $\vec{x}_j$  to a higher dimensional space, and minimizing the objective function implies that we are maximizing  $1/\|w^i\|$ , which is the margin between the two sets of data in  $D$  dimensional space. Since data is not always

linearly separable, we have the penalty term  $C$  that reduces the number of training errors.

After training  $k$  SVM models, each of them produces a decision function:

$$\begin{aligned} & (w^1)^T \phi(x) + b^1, \\ & \vdots \\ & (w^k)^T \phi(x) + b^k \end{aligned}$$

If we need to classify a feature vector  $\vec{x}$ , we compute each of the decision functions, and assign the class label which corresponds to the function with the maximum value.

### 3.2.3 Aspect Category Detection

The objective of this sub-task is to detect the aspect categories expressed in a sentence with respect to a given set of categories (e.g., *food*, *service*, *price*, *ambience*, *anecdotes/misellaneous*) as explained in Chapter 1. Since a sentence can contain several categories, we employ multi-label one-vs-all Support Vector Machines (SVMs) in conjunction with the following vector-based features for a given sentence:

- **Normalized Average Vector (NAV)** is obtained by averaging the vector representations of the words in the sentence. That is, given a sequence of words  $S = w_1, w_2, \dots, w_n$ , the normalized average vector is computed as follows:

$$NAV = \frac{\frac{1}{N} \sum_{i=1}^N v_i}{\left| \frac{1}{N} \sum_{i=1}^N v_i \right|} \quad (3.3)$$

where  $N$  is the number of words,  $v_i$  is the vector representation of  $w_i$  in the sentence, and  $|x|$  means the  $L2$ -norm of  $x$ . In addition, we only consider adjectives, adverbs, nouns, and verbs to compute the NAV. This is because these word types capture most semantic and sentiment information in a sentence.

- **Number of Tokens (TN)** is number of words in sentence that are used to compute NAV. Although NAV is effective for this task, some information like TN is missing during the averaging process.
- **Category Similarities (CS)** are computed for each predefined aspect category. To compute CS, we first identify a set of words (called *seeds*) for each category by selecting top 20 nearest word vectors to the vector of category name, as shown in

Figure 3.5. Then, for each category, we compute the cosine similarity between its seed vectors and the word vectors of the input sentence. We consider the maximum cosine similarity as a feature representing the similarity between the category and the input sentence.



FIGURE 3.5: This figure shows the projection of category representative vectors on two dimensions. Each cluster represents the set of top 20 vectors that represent that corresponding category. Table 3.1 lists all the representative words.

Food			Price		
food	foods	foods-	price	price-	price-tag
delicacies	natural/organic	offerings	pricing	pricetag	expensive
foods'	produce	supermarket	prices	prices-	pricewise
staples	supermarkets	groceries	cost	quality	markup
cuisines	meats	cuisine	price-wise	rate	pricey
foodstuffs	ready-to-eat	foodstuff	rates	priced	\$
items	flours	fruits/veggies	value	price-	mark-up
Ambiance			Service		
ambience	surroundings	cozy	service	servers	-service
ambiance	interior	classy	service-	efficient	attentiveness
atmosphere	atomosphere	decor/atmosphere	servic	polite	exceptionally
decor	atmosphere-	atmoshere	sevice	service-	waiters
environment	decore	atmoshpere	waitstaff	courteous	attentive
vibe	atomsphere	setting	staff	ambience	ambiance
decor	atmostphere	decor/ambiance	consistently	prompt	

TABLE 3.1: This table lists the category representative words for each one of the categories. Figure 3.5 projects the vectors of those words on a two dimensional space.

### 3.2.3.1 Multilabel SVM

The Aspect Category Detection subtask is formulated as a multi-label multi-class classification. We treat this subtask at the entire review level. Given a review, we compute one feature vector  $\vec{x}$ . The output of the classification algorithm is  $y$  where  $y \subset \{Food, Price, Service, Ambiance, Misc.\}$ . Every review can be labeled with one or more category labels. To implement this model, we use a multi-label multi-class SVM.

The algorithm works exactly like the one-vs-all SVM, explained in subsection 3.2.2.1, with the main difference in applying the decision function. First, we train one SVM model for each label (one input can contribute to multiple labels), and when we want to classify a new data point, we compute the decision function for all the possible classes, and if the decision value is greater than some threshold  $\tau$ , we assign  $\vec{x}$  to that class. Thus,  $\vec{x}$  can have more than one class assigned to it.

## 3.3 Extra Features

This section will list a number of features that we tried on the subtasks, but it did not prove to be very helpful. In Chapter 4 we will discuss the impact these extra features had, and try to explain why they were not very successful.

### 3.3.1 Aspect Term Extraction

The aspect term extraction task was tested with some of the features designed for the aspect sentiment prediction, and aspect category detection tasks. For those features, instead of computing a feature relative to an aspect term, it is computed relative to every word in the review. For example, when we apply the ADV feature, we compute the average dependency vector of each word in the review, and use that vector for the feature representation of that word. The same applies to the other features.

Also, some features were computed considering the context as well. For example, we include the POS tag of the word next to the word of interest as another feature. This is indicated by using **left {feature name}** or **right {feature name}**.

Besides the features mentioned before, additional features designed for aspect term extraction are listed here:

- **Dependency Similarity (Dep-CS):** The dependency similarity feature is computed just like the Category Similarity (CS) feature (explained in section 3.2.3),

but instead of using the word’s vector representation, we use the ADV vector to represent the word of interest.

- **Word Class (WC):** We used the Yelp dataset and `word2vec` to cluster the words into 300 clusters. The first step was to compute vector representations of words in the Yelp dataset, then use k-means with  $k = 300$  to cluster the words into 300 different clusters. The hypothesis was that some clusters are more likely to be aspects than others based on the vector representation of the words. Then for every word we use the cluster number as one feature for that word.
- **Binary Word Class (BWC):** This feature is the same as the Word Class feature except that the feature is represented as a binary one-not 300-dimensions vector. For row  $i$  in the vector,  $i = 1$  if the word is in cluster  $i$  otherwise  $i = 0$ . This representation adds extra dimensionality to the feature vector.
- **Conditional POS:** Conditional POS is designed for neighboring POS tags to consider the word’s context. To achieve this, we consider the POS tag of the previous/next word unless that POS tag is a conjunction, preposition, determiner, or pronoun. In that case, we skip that word and get the POS tag of the word before/after it.

### 3.3.2 Aspect Sentiment Prediction

- **Rating Vectors - two (RV-two):** this feature is similar to the RV feature (explained in section 3.2.2), but for RV-two, we only consider two subsets of the data. We consider data that is rated 1 (extremely negative) and data that is rated 5 (extremely positive).
- **Dependency Average All Ratings (ADV-AllRatings):** this feature is obtained just like ADV (explained in section 3.2.2), by averaging the vector representations of the *dependency words* (DW) for the aspect. The difference is that ADV-AllRatings generates 5 different vectors. Each vector is computed using a vector model trained on one of the 5 Yelp ratings subsets, where each subset contains reviews with a specific rating (from 1 to 5).
- **Positive Negative Similarities Difference (PNS-difference):** the PNS-difference feature is computed from the PNS features (described in section 3.2.2). The output of the PNS feature is the maximum positive similarity, and the maximum negative similarity. PNS-difference takes the difference between these two values and considers that as a feature. The idea behind this is to be able to detect reviews with conflict and neutral labels. If a review has both positive and negative sentiments

no	couldn't	isn't	scarcely
not	won't	wasn't	barely
none	can't	shouldn't	'nt
no one	don't	wouldn't	doesn't
nobody	nothing	hardly	never
nowhere	neither		

TABLE 3.2: List of most common negation words.

and the PNS-difference is close to zero, and the overall sentiment will be labeled as conflict.

- **All-PNS:** this feature is computed just like the normal PNS feature, but instead of taking the maximum similarity with negative and positive seeds, it uses all the similarity scores with the seeds as features. So instead of computing which similarity score is larger, instead it includes all of the similarities scores as extra dimensions in the feature vector.
- **PNS-specific:** PNS-specific is computed exactly like the normal PNS feature except that it only considers the similarity for nouns, adjectives, and adverbs.
- **All-PNS-specific:** All-PNS-specific is computed exactly like the normal All-PNS feature except that it only computes the similarity for nouns, verbs, adjectives, and adverbs.
- **Negation:** The negation feature is a binary feature that indicates whether there is a negation term in the word dependency neighborhood (as described in ADV) or not. This feature helps us detect negated sentiments such as “the tea was not good”. To compute this feature, we first generated a list of the most common negation words, as shown in Table 3.2.

Then for each aspect term, we compute the *dependency words* and check if any of them contains one of the negations words.

### 3.3.3 Aspect Category Detection

- **Average Vector (AV):** this feature is similar to NAV which is explained in section 3.2.3 but this one does not have any normalization. Thus, the corresponding equation 3.3 changes to the following.

$$AV = \frac{1}{N} \sum_{i=1}^N v_i \quad (3.4)$$

where  $N$  is the number of words,  $v_i$  is the vector representation of  $w_i$  in the

sentence. In addition, we only consider *adjectives*, *adverbs*, *nouns*, and *verbs* to compute the AV. This is because these word types capture most semantic and sentiment information in a sentence.

- **Similarity Vectors (SV):** these features are four word vectors of the input sentence that are found by computing the CS features (CS is explained in Section 3.2.3). That is, SV indicate the word vectors with the highest cosine similarities with the category seeds.
- **Weighted Similarity Vectors (WSV):** these features are similar to the SV, but each vector is weighted by its respective cosine similarity with the input sentence.

# Chapter 4

## Evaluation and Results

In this chapter, first we will explain the experimental settings and the data used in our experiments. Then, we will discuss the results obtained from our approach on the aspect-based sentiment analysis subtasks.

### 4.1 Task and Data

In the Aspect-Based Sentiment Analysis task, we outlined three subtasks to solve.

1. Aspect Term Extraction: given a user-generated review, label all the aspects in the review. This task is formulated as a single-label sequence tagging task.
2. Aspect Polarity Prediction: given a set of aspects from a review, assign each aspect a sentiment (*positive*, *negative*, *neutral*, or *conflict*). This task is formulated as a classification task.
3. Aspect Category Detection: given a user-generated review, detect the categories the review is discussing. This task is formulated as a multi-label classification task.

#### 4.1.1 Restaurant Review Dataset

To proceed with achieving the tasks summarized above, we need to use a dataset of reviews. We use the restaurant review dataset<sup>1</sup> provided by (Ganu et al., 2009a, Pontiki et al., 2014) that was used in the 2014 SemEval Challenge. The dataset contains 3,041 training and 800 test sentences. The dataset includes annotations of coarse aspect

---

<sup>1</sup>This dataset can be found at the SemEval website: <http://alt.qcri.org/semeval2014/>

Categories	food	srvc.	price	amb.	misc.	Total
Train	1,232	597	321	431	1,132	<b>3,713</b>
Test	418	172	83	118	234	<b>1,025</b>

TABLE 4.1: Category distributions over the dataset.

categories of restaurant reviews. The dataset is also extended to include annotations for aspect terms in the sentences, aspect term polarities, and aspect category-specific polarities. The annotations were performed by experienced human annotators from the SemEval team (Pontiki et al., 2014).

The training dataset contains 3,693 aspects and 3,713 categories, and the test dataset contains 1,134 aspects and 1,025 categories. On average, a review contains 1.23 aspects, Figure 4.1 shows the distribution of number of aspects in a review. In the histogram we notice an exponential drop in the number of aspects in a review. Generally, reviews are focused, and discuss only a small number of aspects. This tells us that there is high correlation between the general sentiments in the review and the sentiments of the specific aspects.

In the dataset, the predefined aspect categories are *food*, *service*, *price*, *ambiance*, *anecdotes/misellaneous*, Table 4.1 shows the distributions of these categories over the dataset, and Table 4.2 shows the distribution of number of categories over reviews. This table also shows that the categories follow the same behaviour as the aspects, which is expected. Most of the reviews only discuss a single category of interest which makes sense, since reviews also only discuss a small number of aspects.

#Cats	Train	Test
1	2465	611
2	486	155
3	84	32
4	6	2

(a)

#Cats	Train (%)	Test (%)
1	81%	76%
2	15%	19%
3	2%	4%
4	0.1%	0.2%

(b)

TABLE 4.2: (a) Number of reviews that contain multiple categories in each subset.

(b) Percentage of reviews that contain multiple categories in each subset.

#### 4.1.2 Word2Vec Datasets

Since we are interested in implementing vector space methods as our main features, we need to use a corpus to train the word2vec model that takes a word and returns its vector representation. Section 3.2 describes the skip-gram model we train for Word2Vec (Mikolov, 2014, Mikolov et al., 2013a,b,d). The model needs a large amount of unstructured text data for training the word vector representations.

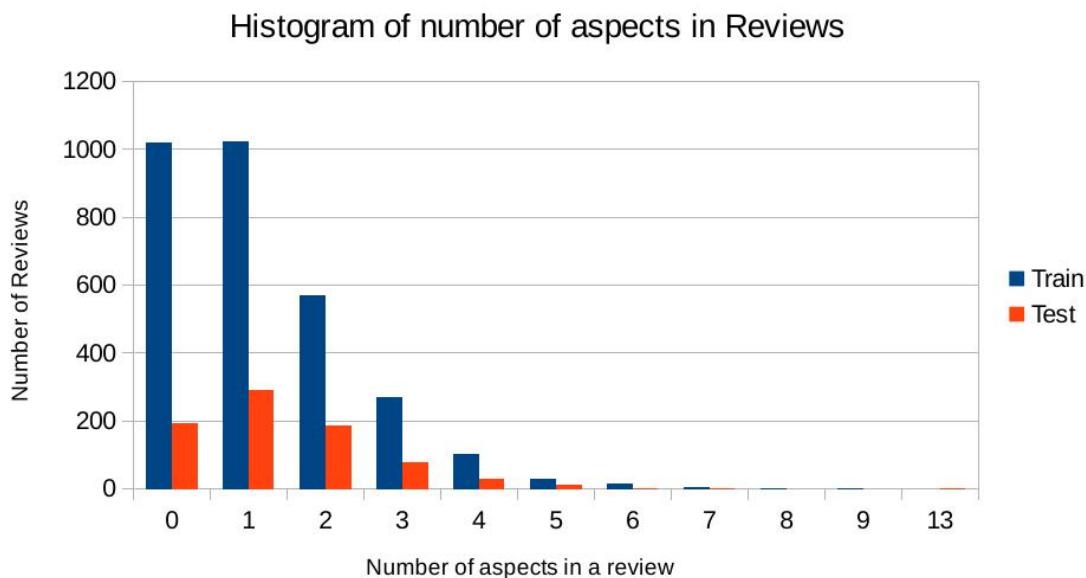


FIGURE 4.1: The plot shows the distribution of number of aspects in a review in both the training and test datasets. In total, the training dataset has 3,693 aspects and the test dataset has 1,025 aspect terms.

We used two datasets for training, first is the GoogleNews dataset (Mikolov, 2014) that contains 3 million unique words and about 100 billion tokens. To account for the effect of domain information on the quality of word representations, we also use a dataset of restaurant reviews<sup>2</sup> from Yelp that contains 131,778 unique words and about 200 million tokens.

## 4.2 Experimental Platform

To evaluate the vector-based features we developed and described in Chapter 3, we developed a platform in JAVA built on top of uimaFIT (Ogren and Bethard, 2009), dkpro-core (de Castilho and Gurevych, 2014), and dkpro-tc (Daxenberger et al., 2014). UIMA is a framework architecture built to support analysis of unstructured data (Ogren and Bethard, 2009). It defines interfaces and guidelines for designing analysis components as well as easy access to libraries of recyclable components. Each component is called an Analysis Engine (AE), and takes input text and performs a specific analysis task on it. In our pipeline, the cycle starts at a Reader that process the input data, then passes that to the Annotators in order to perform pre-processing on the text. The Annotators pass the output to a Feature Extractor that computes a feature vector from the annotated text. Finally the feature vectors are passed to a classifier for training or testing. The process is illustrated in Figure 4.2.

<sup>2</sup>This dataset is available on: [http://www.yelp.com/dataset\\_challenge](http://www.yelp.com/dataset_challenge).

### 4.2.1 Readers

The first step was to digest the review data. To do so we developed Readers (as named by dkpro-tc). The Reader parses through the input data and annotates the text and labels into a UIMA jCas to have it accessible by the rest of platform (a good tutorial for UIMA is (Ogren and Bethard, 2009)). jCas is a UIMA data structure that holds the raw data as well as all the annotations added to it, it's used to pass the information from one UIMA component to the other. The input data was in XML format, so we developed a generic XMLReader, and used it throughout all the different subtasks annotating different elements as the labels. For example, in the aspect extraction task we extract the aspect information as the label where in the category detection task we extract the category information as the label.

### 4.2.2 Annotators

After the data is loaded in the UIMA framework, dkpro-tc applies the pre-processing step using the UIMA Annotators. An Annotator will manipulate the data in the jCas and can add extra annotation to it. For the Annotators, we used the Stanford POS tagger, the Stanford Segmente, the Stanford Lemmatizer, and the Stanford Parser from the dkpro-core library (see Figure 4.2).

One simple example for how the annotators work is the Stanford POS tagger. For every word in the jCas, it will annotate it with its POS tag. After that annotator is applied, every word in the jCas is annotated with its POS tag.

### 4.2.3 Feature Extractors

The next step is computing feature vectors from the processed data. To compute a feature we develop a Feature Extractor that takes in the annotated data and outputs a feature vector (often numeric). Depending on the subtask, Feature Extractors operate either at the Unit level (word level) or the Document level (review level). For each of the 21 features described in Chapter 3, we developed a Feature Extractor that computes that feature.

The output of all Feature Extractors is then passed to generate a single feature vector for each input  $w_i$ . Those feature vectors are passed to the classifiers, depending on the subtask, to perform the training, testing, or cross-validation (see Figure 4.2).

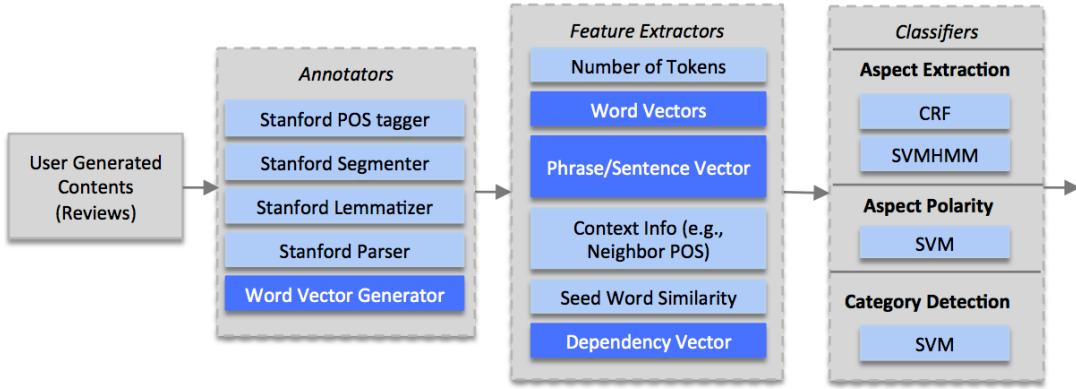


FIGURE 4.2: The figure illustrates the work-flow of the dkpro-tc application we developed. In our pipeline, the cycle starts at a Reader that process the input data, then passes that to the Annotators in order to perform pre-processing on the text. The Annotators pass the output to a Feature Extractor that computes a feature vector from the annotated text. Finally the feature vectors are passed to the classifier.

#### 4.2.4 Classifiers

A number of the classification algorithms used to train and test our model needed some parameter tuning. This section will describe the process used in picking the parameter value for each of those algorithms.

**Aspect Term Extraction (Task 1)** For the Aspect Term Extraction task, one of the classifiers we used was the Conditional Random Field (CRF) classifier. The CRF does not have any tuning parameters, but there are a number of different ways to train a CRF.

- **arow:** Adaptive Regularization Of Weight Vector minimizes the loss  $s(x, y') - s(x, y)$ , where  $s(x, y')$  is the score of the Viterbi label sequence, and  $s(x, y)$  is the score of the label sequence of training data (Mejer and Crammer, 2010).
- **lbfgs:** Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method maximizes the logarithm of the likelihood of the training data with L1 regularization terms (Nocedal, 1980).
- **l2sgd:** Stochastic Gradient Descent (SGD) with L2 regularization maximize the logarithm of the likelihood of the training data with L2 regularization term(s) using Stochastic Gradient Descent (SGD) (Shalev-Shwartz et al., 2011).

To choose the best training algorithm, we implemented five-fold cross validation using the POS tag feature and the 300 dimension vector representation of a word. In Table

Training Algorithm	Precision	Recall	F1
arow	90.88	73.80	81.45
<b>lbfsgs</b>	<b>92.43</b>	<b>76.51</b>	<b>83.72</b>
l2sgd	92.38	75.24	82.93

TABLE 4.3: This table shows the results of using three different algorithms to train the CRF. Each CRF model was trained on the entire training set using 5-fold cross validation and using two types of features, the POS tag and the word vector representation.

From this table, we see that lbfsgs algorithm has the best performance.

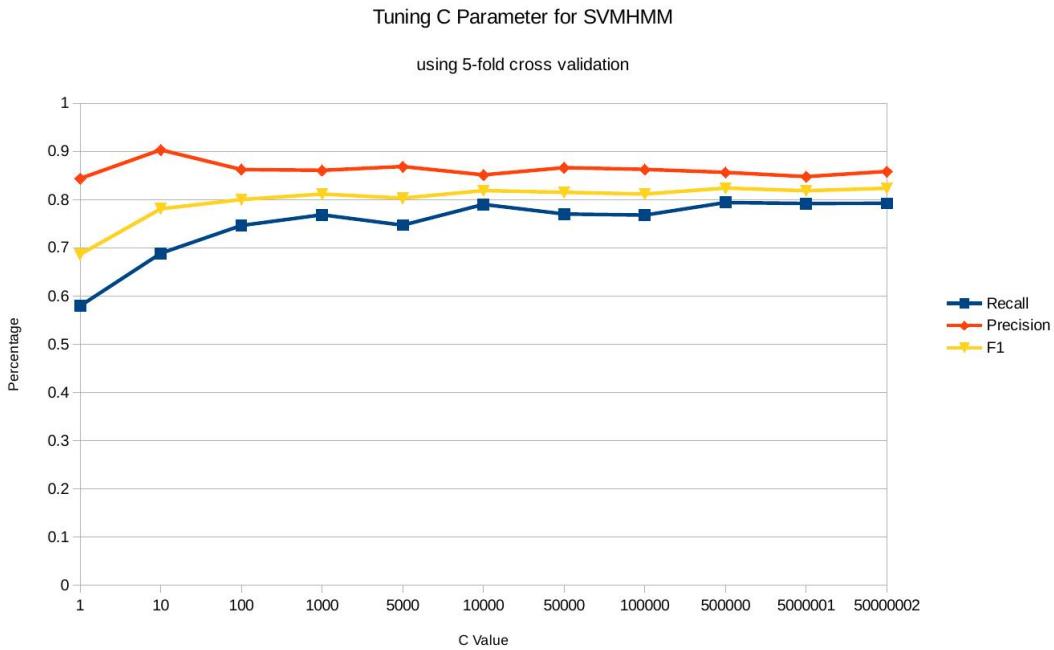


FIGURE 4.3: This plot shows the recall, precision, and F1 values for different values of the C parameter for the SVM-HMM aspect term extraction model.

4.3 we see the average Precision, Recall, and F1 for the different training algorithms. From the results table, we can see that the **lbfsgs** algorithm performs better than the other two. Thus it was chosen moving forward with CRF.

We also tuned the  $C$  parameter in the SVM-HMM.  $C$  is a parameter to trade off margin size and training error. To tune the  $C$  parameter, we also implemented a five fold cross validation using the POS tags and word vector representation features. Figure 4.3 shows the precision, recall, and F1 values for different values of  $C$ . Based on these results, we observe a wide range of possible  $C$  values without effecting the results significantly. We set  $C = 100,000$  to obtain the best results and not over penalize the model.

**Aspect Sentiment Prediction (Task 2)** In the Aspect Sentiment Prediction sub-task, we used the one-vs-all SVM algorithm to train the prediction model. As described in section 3.4, the one-vs-all SVM has a parameter  $C$  that trades off the size of the

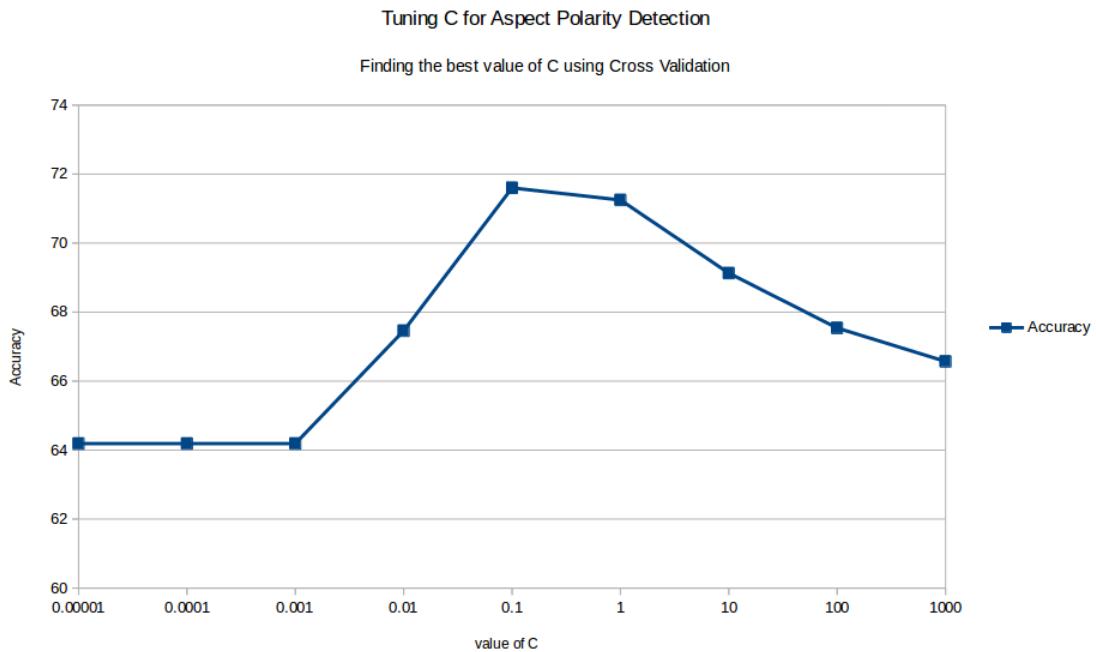


FIGURE 4.4: This plot shows the average accuracy for 5-fold cross validation for every  $C$  value. In all the experiments we used the Average Dependency Vector (ADV), and the Rating Vectors (RV) as the features. We set  $C = 0.1$

margin in the classifier with training error. To tune that parameter, we used a five-fold cross-validation experiment to evaluate each  $C$  value. For each of the experiments we used the Average Dependency Vector (ADV), and the Rating Vectors (RV) features. Figure 4.4 shows the average accuracy for each cross validation experiment for a specific  $C$  value. From that result, we set  $C = 0.1$ .

**Aspect Category Detection (Task 3)** For the Aspect Category Detection subtask, we used the Multilabel SVM with a tunable  $C$  parameter, described Section 3.5.  $C$  is the parameter that sets the trade off between the margin size in the SVM and the training error. To tune this parameter, we used a five-fold cross-validation on the training set. For the cross validation task, we used three features, the number of tokens (TN), Category Similarities (CS), and Normalized Average Vector (NAV). The average of the cross validation results is reported in Figure 4.5. As we can see, the performance is not very sensitive to change in  $C$  for values between 0.15-0.55. Any  $C$  in the specified range is expected to perform similarly, to move forward we picked  $C = 0.25$

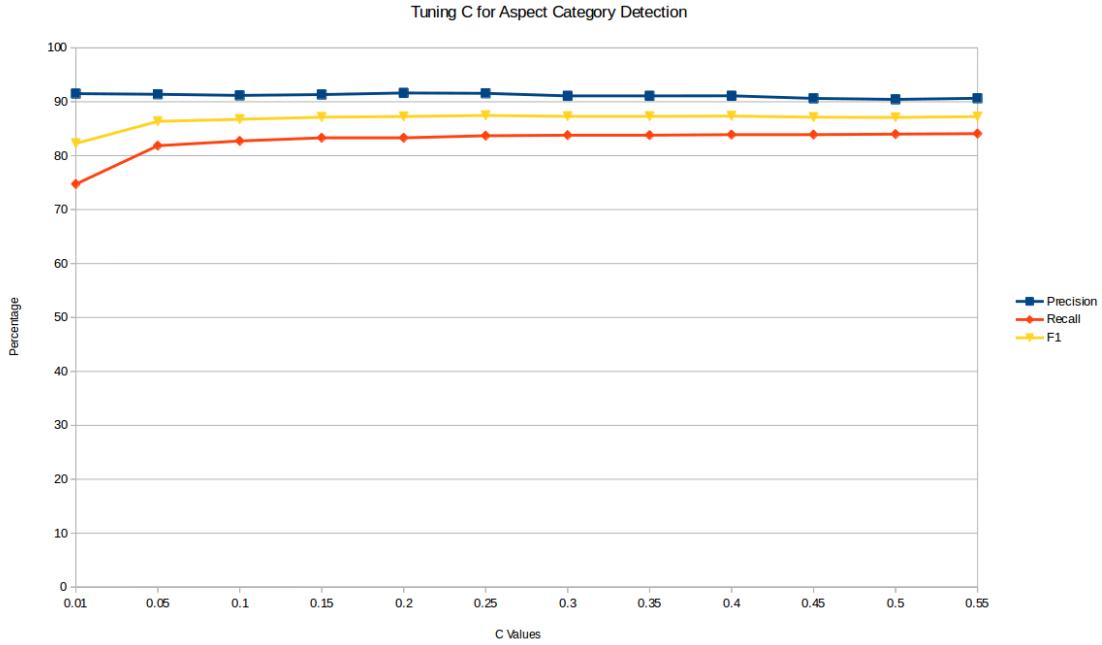


FIGURE 4.5: Results for 5-fold cross validation for tuning C in Category Detection subtask. We are plotting the average of the 5 experiments using that specific value of  $C$ . We used the number of tokens, category similarities, and normalized average vector as the features. There is a wide range for possible  $C$  values, we set  $C = 0.25$

### 4.3 Task 1: Aspect Term Extraction

The objective of Aspect Term Extraction is to identify the aspect terms appearing in the restaurant review text. For restaurants, the aspect terms can be burgers, pizzas, the music played, temperature, etc. This task is formulated as a sequence tagging tasks where every word in the review gets tagged with either aspect or not-aspect.

#### 4.3.1 Evaluation Metrics

For aspect term extraction subtask, we will be using two evaluation metrics (1) *Aspect-Level Evaluation*, (2) *Word-Level Evaluation*.

- **Aspect-Level Evaluation:** this metric is based on evaluating how many aspect terms we get correct. Using the known definitions of Precision ( $P$ ), Recall ( $R$ ), and  $F1$  scores:

$$P = \frac{|S \cap G|}{|S|}, R = \frac{|S \cap G|}{|G|}$$

$$F1 = \frac{2}{\frac{1}{P} + \frac{1}{R}} = \frac{2PR}{P+R}$$

$S$  will be the set of all predicted aspect terms where multi-word aspects compromise a single element in the set.  $G$  then will be the gold standard (correct) annotations for the aspect terms where again multi-word aspects are a single element in the set.

- **Word-Level Evaluation:** this metric looks at individual words in a review and evaluates how many of them were labeled correctly. Using the same definition for Precision and Recall,  $S$  the set of all words labeled as “Aspect” and  $G$  will be the gold standard of all words that are labeled “Aspect”. In this case, multi-word aspects will result in multiple elements in the set where each word is an element.

From the definition of these two evaluation metrics, we can see that the Aspect-Level metric is more strict than the Word-Level. If we predict 3 out of 4 words of an aspect term, the first metric will mark the entire prediction as incorrect where the second metric will mark 3 words as correct and one as incorrect. Even when we predict more words than the aspect has, it’s evaluated in a similar way.

It is important for us to look at those two metrics because we can learn from the difference between them. For example if both metrics are low, we know that our model is not doing a good job identifying where the aspects are within a review. If the Aspect-Level is low, but the Word-Level is high, we know that we are able to capture where the aspects are in the review, but we are not capturing the entire sequence of words in a multi-word aspect. This is important, because about 30% of the reviews in the dataset have a multi-word aspect. For example, the phrase “I liked their hot fries” contains the multi-word aspect “hot fries.”

### 4.3.2 Aspect Term Extraction Results

The results of our vector-based approach for this task are shown in Table 4.4 and Table 4.5. The first cell of Table 4.4 shows the F1 performance of 47.15% produced by our baseline. The baseline creates a dictionary of aspect terms from the training data, and then a given sequence of words are tagged as aspects by looking up the dictionary (Pontiki et al., 2014). This approach cannot handle the out of vocabulary aspects.

As explained in Section 3.2.1, we employ CRFs and SVM-HMM for this task. As features we utilize POS-tags of words, and vector representations computed by `word2vec` and trained on Yelp (Y300) or GoogleNews (G300) data to generate 300 dimensional vectors. The corresponding results are shown in the last two rows of Table 4.4. These results indicate the vector representations trained on Yelp data leads to a high performance for both CRF and SVM-HMM. The GoogleNews dataset contains a larger vocabulary of

around 3M, words as compared to the Yelp data with around 100K words. This indicates the effectiveness of the domain in the quality of word representations.

<b>Baseline-F1 = 47.15</b>	CRF Suite			SVM-HMM		
Features	Precision	Recall	F1	Precision	Recall	F1
POS-tags	44.09	9.87	16.13	44.58	9.43	15.57
POS-tags + word2vec (Y300)	82.38	72.57	77.16	77.33	78.83	<b>78.08</b>
POS-tags + word2vec (G300)	82.69	74.16	<b>78.19</b>	76.88	74.51	75.68

TABLE 4.4: Aspect-Level Evaluation results for the aspect term extraction task.

	CRF Suite			SVM-HMM		
Features	Precision	Recall	F1	Precision	Recall	F1
POS-tags	64.34	22.17	32.97	67.18	20.96	31.95
POS-tags + word2vec (Y300)	92.55	74.82	82.74	90.17	82.34	<b>86.08</b>
POS-tags + word2vec (G300)	92.43	76.51	<b>83.72</b>	90.34	78.91	84.24

TABLE 4.5: Word-Level Evaluation results for the aspect term extraction task.

To evaluate the effectiveness of the vector-based features, we repeat our experiments with only the POS-tags feature. The performance dropped significantly, as shown in the table. Although nouns can be strong candidates for aspects (Blinov and Kotelnikov, 2014), the majority of aspects, like multi-word aspects cannot be captured by only considering their POS-tags.

As Table 4.5 shows, the performance at the Word-Level increases by more than 8% compared to the Aspect-Level metric, which means we are detecting more aspect terms but the aspect-level evaluation is disregarding them for being incomplete aspects. We could potentially add features that focus on capturing the correct length of the aspect term based on the sentence context improve both metrics. The best performing system for aspect term extraction on the same data set was reported by (Brun et al., 2014) with an F1 score of 83.98 on the Aspect-Level evaluation. That is a 5% increase over our best performance. It is important to note though that they use very specific linguistic and lexical features

Figure 4.6 is one output example of the aspect term extraction model applied to one of the reviews from the test set. This example illustrates the point about our model being accurate with detecting the location of the multi-word aspect terms but in many cases it does not perform well in labeling all the words in that aspect correctly. In Figure Figure 4.6 we can see this effect in the aspect term “spicy tuna roll” where the model correctly identifies “tuna roll” as an aspect but misses “spicy” in the labeling.

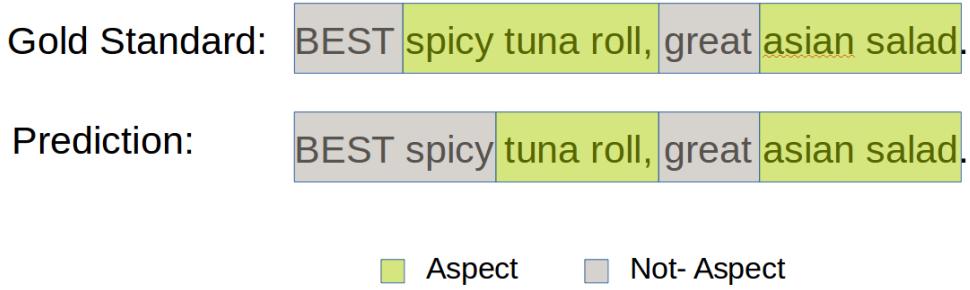


FIGURE 4.6: This figure shows the aspect term extraction model applied to the phrase “BEST spicy tuna roll, great asian salad.” We can see that our prediction model was able to detect both multi-word aspects, but for one of them it did not label all the words correctly.

## 4.4 Tasks 2: Aspect Sentiment Prediction

The objective of the Aspect Sentiment Prediction task is to classify each aspect to one of the sentiment classes  $\{positive, negative, conflict, neutral\}$ . This task is formulated as a multi-class classification task, where each aspect gets assigned to one out of the four possible sentiment classes. We use the aspect terms from the gold standard in this subtask.

### 4.4.1 Evaluation Metric

To evaluate the performance of our models on this task, we calculated the accuracy of the model prediction. Accuracy is defined as the number of correctly predicted aspect term sentiments divided by the total number of aspect terms. We use the gold standard annotations to check if we predicted a sentiment correctly.

### 4.4.2 Aspect Sentiment Prediction Results

The first cell of Table 4.6 shows a performance of 64.28% obtained by our baseline. The baseline tags a given aspect in a test sentence by the most frequent sentiment for the aspect in top  $K$  similar training sentences to the test sentence. In addition, for the out of vocabulary aspects (ones that were not encountered in training data), the majority sentiment over all aspects in training data will be assigned. (Pontiki et al., 2014)

The results of our approach for this task are shown in Table 4.6. The SVMs are applied to this task and the parameter  $C$  for SVMs is optimized through cross-validation on training data, as explained in Section 4.1.1. The third row of the table shows the results

<b>Baseline-Accuracy = 64.28</b>		SVM (C = 0.1)			
Features		Pos-F1	Neg-F1	Neu-F1	Accuracy
ADV (Y300)		82.70	52.30	31.39	71.34
RV (Y300)		83.26	51.79	32.85	71.95
RV + PNS (G300)		83.48	53.29	32.97	<b>72.39</b>

TABLE 4.6: Results for the aspect sentiment prediction task.

when we use the *Average Dependency Vector* (ADV) computed based on `word2vec` trained on all the Yelp (Y300) data. As explained in Section 3.2.2, to investigate the distribution of words (Amiri and Chua, 2012) and their vector representations over different ratings, we present *Rating Vectors* (RV). RV features include 4 ADVs in which four vector representations for a word are computed on Yelp reviews with ratings 1, 2, 4, and 5, respectively. Reviews with the rating 3 are not considered, because they are mostly of neutral or conflict orientation. Using RV results in a better performance, as shown in the fourth row of Table 4.6. However, there is not a significant difference between the results of experiments with RV and ADV. The reason is that most of the reviews in the Yelp data have positive ratings (i.e., ratings 4 and 5) and as such the distributions of words does not dramatically changed as compared to the whole review data.

The highest performance is achieved when we use the combination of RV and *Positive/Negative Similarities* (PNS) features, as shown in the fifth row of the Table 4.6. Since the vector representations for some positive and negative words (e.g., good and bad) are similar, PNS feature provides more information for a classifier to distinguish between these vectors by defining a set of positive and negative vectors, as explained in Section 3.2.2.

The best performing system for aspect sentiment prediction using the same restaurant data set is reported by (Wagner et al., 2014). They achieved an accuracy of 80.95% which is 8% higher than our best performance. The best performing system uses a manually designed rule based approach to achieve their result. While our system is less accurate is uses simpler features that are more generalizable.

Figure 4.7 is one output example of the aspect polarity prediction model applied to one of the reviews from the test set. The figure shows two reviews, the first review our model is able to predict the sentiment correctly, as for the second review, the model fails to predict the sentiment of the word “menu”. This is because the model we trained is biased toward positive sentiments since the training data set has very little instances of neutral or conflict aspects.

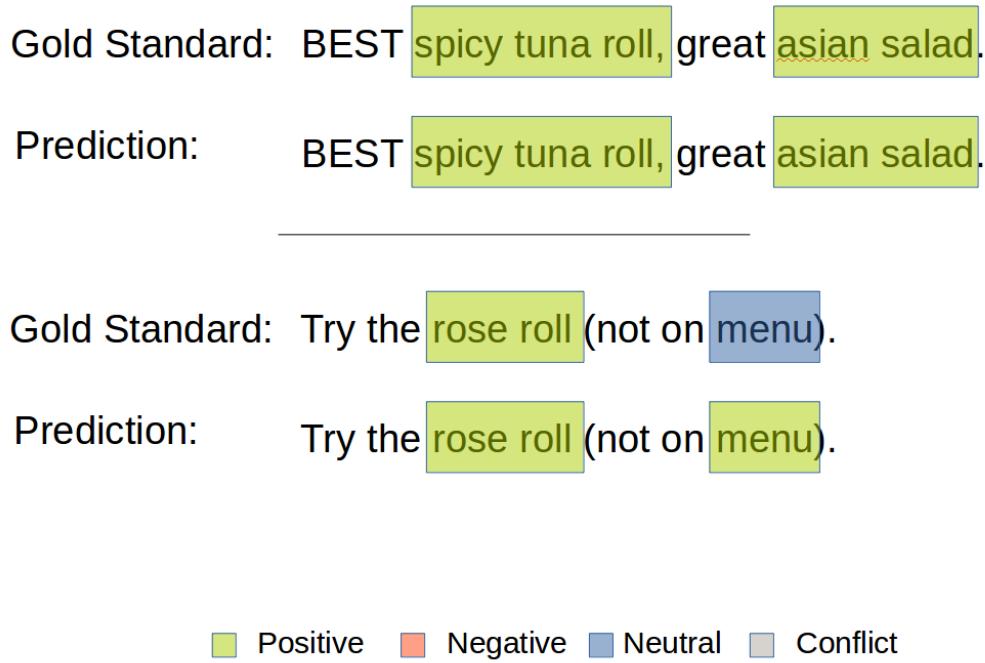


FIGURE 4.7: This figure shows the aspect polarity prediction model output on reviews from the test data set. In the second example we can see that our model is biased towards positive sentiments.

## 4.5 Task 3: Aspect Category Detection

The objective of the Aspect Category Detection task is to label each review with all the categories discussed in that review. Aspect Category Detection is formulated as a multi-class multi-label classification task on the review-level, for every review we label with one or more of the following labels { *Food*, *Price*, *Ambiance*, *Service*, *Misc.* }.

### 4.5.1 Evaluation Metric

Aspect Category Detection is evaluated the same way as the Aspect Term Extraction task. We compute the Precision, Recall, and F1 scores based on the ratio between correctly classified labels and the set of predictions and the gold standard, respectively, as formulated in the Aspect Term Extraction Section.

### 4.5.2 Aspect Category Detection Results

The first cell of Table 4.7 shows an F1 performance of 65.65% obtained by the baseline (Pontiki et al., 2014). Given a sentence, the baseline first retrieves a set of  $K$  similar sentences from the training data. The similarity of two sentences is then determined

<b>Baseline-F1 = 65.65</b>		SVM ( $C = 0.1$ )		
Features		Precision	Recall	F1
NAV (Y300)		89.02	80.68	84.64
NAV + TN (Y300)		90.42	81.07	85.49
NAV + TN + CS (Y300)		91.18	82.73	<b>86.75</b>
NAV + TN + CS (G300)		91.51	81.07	85.98

TABLE 4.7: Results for the aspect category detection task.

by computing the Dice Coefficient between the sets of distinct words in the two sentences (Pontiki et al., 2014). Finally, the input sentence is tagged by the most frequent aspect categories appeared in the  $K$  retrieved sentences. The limitation of this approach is that it employs the text-based similarity measure to measure the semantic similarity between the sentences. However, the results in the table shows that the vector-based features can better capture the semantic similarity between the sentences as compared to the text-based features.

The results of our vector-based approach for this task are shown in Table 4.7. As explained in Section 3.2.3, SVMs are applied to this task with a combination of *Normalized Average Vector* (NAV), *Token Numbers* (TN) and *Category Similarities* (CS) features for a given sentence. These features employ the `word2vec` trained on Yelp (Y300) or GoogleNews (G300) to obtain the vector representations. Their corresponding results are shown in the 5<sup>th</sup> and 6<sup>th</sup> rows of the table. The results imply the impact of our vector-based features that lead to the highest performance using the Yelp data.

To evaluate the effectiveness of above vector-based features, we repeat our experiments with different combinations of those features. Lower performance is achieved by using NAV and TN and ignoring the CS, as shown in the 4<sup>th</sup> row of Table 4.7, and by using NAV and ignoring both CS and TN, as shown in the 3<sup>rd</sup> row of the table.

Best performing system for aspect category detection using the same restaurant data set is reported by (Zhu et al., 2014). They achieved an F1 score of 88.57% which is 2% higher than our best performance. The best performing system uses a set of lexicon features, Ngrams, negation features, POS tags, and clustering features.

Figure 4.8 shows an example category prediction for two reviews from the test data set. The first review “Also, the staff is very attentive and really personable.” talks about service, and our model correctly predicted that. The mention of the word “staff” in a review is a very good indicator for reviews that discuss the service of the restaurant and our model learnt that successfully from the training data. The other example shows a review where our model did not predict the correct categories. The model predicted the category *Ambiance* for having the word “feel” in the review. Usually, when reviewers

discuss how they feel it is often with regards to the ambiance, for example “the lighting feels very romantic”.

**Review:** Also, the staff is very attentive and really personable.

**Categories (Gold Standard):** {Service}

**Categories (Prediction):** {Service}

---

**Review:** From the moment you walk in, you feel like you're in the perfect place.

**Categories (Gold Standard):** {Misc.}

**Categories (Prediction):** {Ambiance, Misc.}

FIGURE 4.8: This figure lists two example reviews from the test data set. For the first one, the model predicts the correct category of *Service* by seeing the word “staff” in the review. In the second one, the model mistakenly predicts the category *Ambiance* because of the word “feel” in the review.

## 4.6 Extra Features Results

In the previous subsections we presented the results for the best performing features. In this subsection we will go over the results for the other features we tried in each of the subtasks.

### 4.6.1 Aspect Term Extraction-Extra Features

In this section, we are listing the results of using a number of different features to perform the aspect term extraction task. These features did not improve the performance of the algorithm. The results are reported using the CRF on extra features and the results are presented in Table 4.8. For a description of the features in the table, please refer to Chapter 3.

Features	Precision	Recall	F1
POS	64.34	22.17	32.97
POS + WV(G300)	92.43	76.51	83.72
POS + WV(Y100)	91.85	74.7	82.39
POS + WV(Y200)	92.68	74.7	82.72
POS + WV(Y300)	92.55	74.82	82.74
POS + W2V(G300) + ADV	89.45	73.55	80.73
POS + W2V(G300) + CS + Dep-CS	92.34	74.82	82.66
POS + WV(Y300) + WordClass	92.66	74.52	82.6
POS + WV(Y300) + BinaryWordClass	92.99	74.28	82.59
POS + WV(Y300) + CS	92.55	74.88	82.78
POS + WV(Y300) + CS + WordClass	92.68	74.76	82.76
POS + WV(Y300) + CS + BinaryWordClass	92.79	74.46	82.62
POS + WV(Y300) + left WV(Y300) + right WV(Y300)	88.77	74.28	80.88
POS + right POS + WV(Y300) + CS	92.71	74.34	82.51
POS + WV(Y300) + CS + left Conditional POS	92.04	75.9	83.2
POS + WV(Y300) + CS + right Conditional POS	92.77	74.22	82.46
POS + WV(Y300) + CS + left Conditional POS + right Conditional POS	92.18	75.24	82.85
POS + right POS + WV(Y300) + right WV(Y300) + CS	90.4	74.88	81.91

TABLE 4.8: Results from testing extra features on the aspect term extraction subtask.

#### **4.6.2 Aspect Sentiment Prediction-Extra Features**

In this section, we list the results of using a number of different features to perform the Aspect Sentiment Prediction task. These features did not improve the performance of the algorithm. All the extra features were tested on the SVM algorithm described in section 3. All the results are listed in Table 4.9. For a description of the features in the table, please refer to Chapter 3.

Features	Negative			Neutral			Positive			Conflict			Accuracy
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	
ADV (Y300)	59.87	46.43	52.3	55.13	21.94	31.39	74.72	92.58	82.27	50	7.14	12.5	71.34
ADV (Y300) + PNS	60.54	45.41	51.9	54.32	22.45	31.77	74.7	92.86	82.79	0	0	0	71.34
RV (Y300) + PNS (G300)	64.49	45.41	53.29	58.44	22.96	32.97	74.84	94.37	83.48	0	0	0	72.39
RV (Y300) + PNS + Negation	62.94	45.92	53.1	58.44	22.96	32.97	74.92	93.96	83.36	0	0	0	72.22
RV (Y300) + PNS	64.49	45.41	53.29	58.44	22.96	32.97	74.84	94.37	83.48	0	0	0	72.39
RV-two (Y300)	63.78	41.33	50.15	57.58	19.39	29.01	73.4	94.78	82.73	0	0	0	71.34
RV (Y300)	62.14	44.39	51.79	57.69	22.96	32.85	74.75	93.96	83.26	0	0	0	71.95
ADV-AllRatings (Y300)	62.07	45.92	52.79	55.95	23.98	33.57	75	93.13	83.09	0	0	0	71.86
PNS + ADV-AllRatings (Y300) + PNS-difference	62.5	45.92	52.94	55.95	23.98	33.57	75.03	93.27	83.16	0	0	0	71.95
PNS-specific + ADV	59.44	43.37	50.15	53	27.04	35.81	75.42	91.9	82.85	25	7.14	11.11	71.25
All-PNS + ADV	61.33	46.94	53.18	57.14	24.49	34.29	74.94	92.45	82.78	50	7.14	12.5	71.78
All-PNS-specific	66.27	28.06	39.43	0	0	0	68.32	98.63	80.72	0	0	0	68.16
All-PNS	68.07	41.33	51.43	0	0	0	70.15	97.8	81.7	0	0	0	69.92
All-PNS + ADV	58.6	46.94	52.12	54.32	22.45	31.77	75.17	92.31	82.86	0	0	0	71.25

TABLE 4.9: Results for Aspect Sentiment Prediction subtask using extra features.

### 4.6.3 Aspect Category Detection-Extra Features

In this section, we are listing the results of using a number of different features to perform the Aspect Category Detection task. These features did not succeed to improve the performance of the algorithm. All the extra features were tested on the SVM algorithm described in Chapter 3. All results are listed in Table 4.10. For a description of the features in the table, please refer to Chapter 3.

Features	Precision	Recall	F1
AV(Y100) + TN	88.23	77.56	82.55
AV(Y200) + TN	88.56	80.09	84.11
AV(Y300) + TN	89.66	80.39	84.77
AV(Y300) + TN + CS	90.94	82.34	86.43
AV(Y300) + TN + CS + SimVectors	85.67	81.07	83.3
NAV(Y300)	89.02	80.68	84.64
NAV(Y300) + TN	90.42	81.07	85.49
NAV(Y300) + TN + CS	91.18	82.73	86.75
NAV(Y300) + TN + CS(G300)	91.51	81.07	85.98
NAV(Y300) + TN + CS + WeightedSimVecs	86.72	82.24	84.42

TABLE 4.10: Results for Aspect Category Detection subtask using extra features.

## 4.7 Discussion of Results

In Chapter 4, we described the restaurant data set that was used to train and test the ABSA system. We followed that with a description of the experimental platform developed in Java to train and test the models for the three tasks: Aspect Term Extraction, Aspect Polarity Prediction, and Aspect Category Detection. For each one of these tasks, we used a machine learning model to perform the task. For each model, we tuned its specific parameters using cross-validation.

In the Aspect Term Extraction task, we were able to achieve an *F1* score of 78.19 compared to the state of the art score of 83.98. Our system scores 5 points less than the state of the art, but uses significantly simpler and more generalizable features. In the system we developed, only two features were used. The first feature was the parts-of-speech tag of the word, and the second feature was the word representation in 300 dimensional vector space using the `word2vec` model. The state of the art system uses manually crafted linguistic and lexical features. In the model we developed, we also tried

a number of extra features to improve the performance. The most important of which was to include more information about the context of the word. We tried including the POS tag of the previous word and the following word, as well as include the entire vector representation of the two surrounding words. There was a slight, non significant, improvement when we include the POS tag of the previous word. Other attempted features did not have any significant increase on performance.

For the Aspect Sentiment Prediction task we achieved an accuracy score of 72.39. Our model performs best on predicting positive sentiments, followed by negative ones, then neutral sentiments, which are the hardest to predict. When we compare our results to the state of art performance, we do worse by 8 percentage points in accuracy. The state of the art system used a rule based approach to predict the sentiments, which can be hard to generalize to other domains. In our approach, we used two vector space features that are much easier to generalize.

As for Aspect Category Detection task, we achieved an  $F1$  score of 86.75, which is only 2 percentage points lower than the state of art system (reported  $F1$  score of 88.57). The state of art system uses a set of lexicon features, Ngrams, negation features, and clustering features to achieve their score. All features are text-based. In our vector-based feature set we are able to achieve very similar performance with a lower complexity feature set. We use the vector representation of the review, as well as a cosine similarity to seed words in each category. We would also like to note that when we use the vector representation of only the review, we achieve an  $F1$  score of 84.64, which is still comparable to the state of the art performance. This is good evidence that a vector space representation is capable of capturing the semantic information of user opinions.

In summary, even though none of our tasks achieved better performance than the state of the art, we were able to build an ABSA system that performs within 10% of the state of the art systems, while using vector-based features. The biggest advantage over methods reported in the state of the art work is generalizability. None of the features we compute are specific to the restaurant domain. Thus, vector-based ABSA method can be applied to any other user-generated opinion content. Moreover, since vector-based representations are potentially complementary to the existing state-of-the-art methods, it is possible that system fusion could improve overall results. This will be left as future work.

## Chapter 5

# System Demonstration

In order to demonstrate Aspect Based Sentiment Analysis, we built an online tool to visualize the system we have developed. A snapshot of the demonstration is shown in Figure 5.1.

In the demo, we highlighted three sections, just as the three subtasks, *Aspect Term Extraction*, *Aspect Sentiment Prediction*, and *Category Detection*.

In the first section of the demo we can see the review that is being investigated (i.e. “The selection of food is excellent (I’m not used to having much choice at restaurants), and the atmosphere is great.”) and the aspect terms are highlighted (we see two aspects, selection of food, and atmosphere). The color of the highlighting also corresponds to the sentiment of the aspect term as predicted by the second subtask (e.g. all aspects are positive) as explained in Section 3.2.2. In the Sentiment Prediction section, we display the dependency tree for the review. Each node is a word of the review and in the tree we can easily see the dependency neighborhood for aspect terms that is used to compute the features. The aspect nodes are highlighted with the color of their sentiment. Then in the last section, we display a bar chart of confidence distribution over all the categories. If the confidence is above some threshold, we assign the category to the review (marked in blue), as explained in Section 3.2.3 (e.g. the categories are predicted as food and ambiance in the example shown in the snapshot).

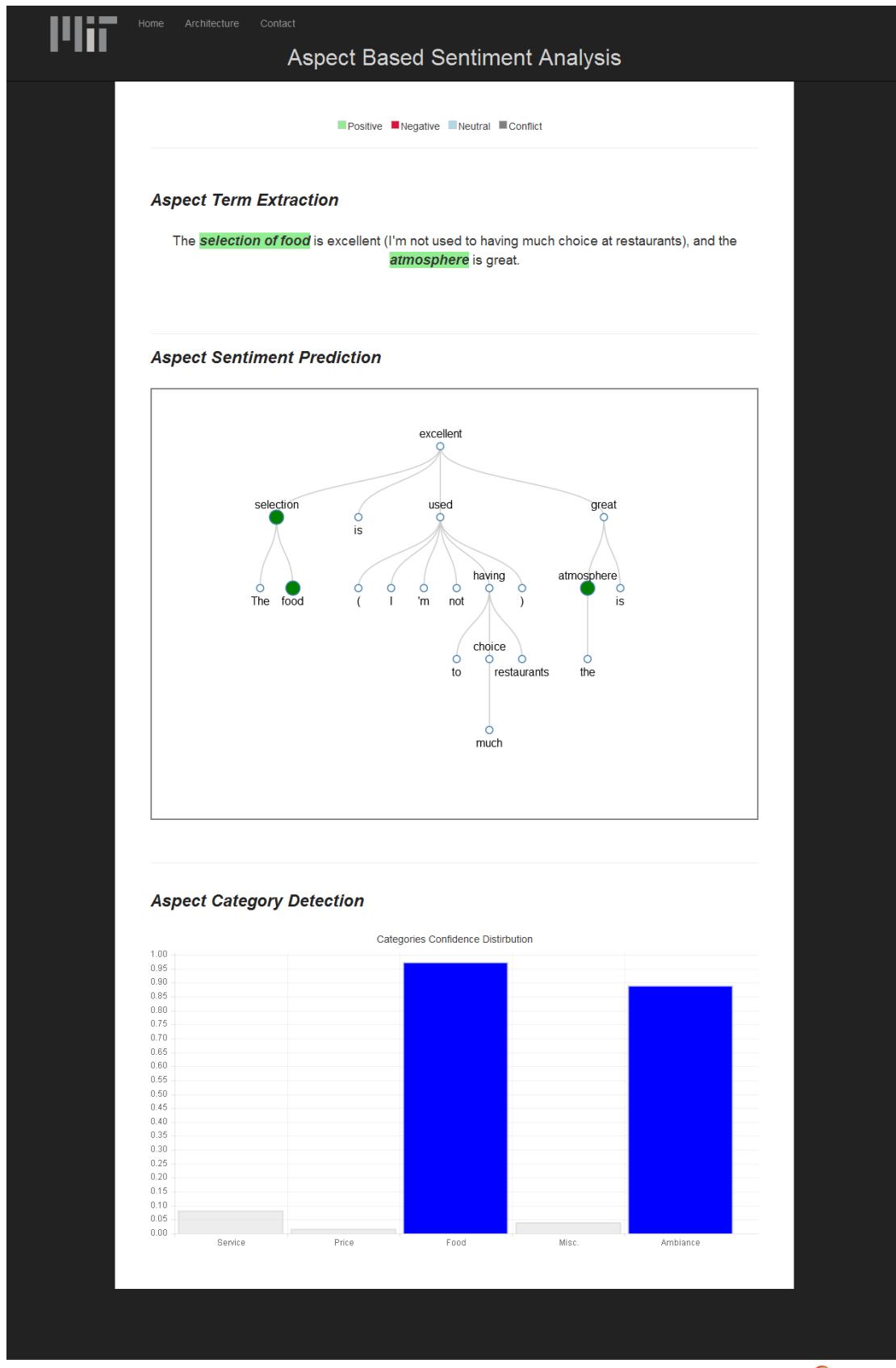


FIGURE 5.1: Screenshot of the online system demo.

# Chapter 6

## Conclusion

The goal of this thesis is to investigate the effectiveness of a vector space representation for Aspect Based Sentiment Analysis, in which we capture both semantic and sentiment information encoded in user-generated content. The ABSA task was broken down into three subtasks: (1) Aspect Term Extraction, (2) Aspect Sentiment Prediction, and (3) Aspect Category Detection.

The ABSA approach addresses a number of limitations faced by traditional sentiment analysis techniques. Mainly it solves the problem of review-level sentiment prediction that can easily overlook critical sentiment information at the aspect level. Our interest was to apply word vector space representation methods to compute vector-based features that can capture both semantic and sentiment information.

### 6.1 Summary

We first developed an experimental platform in JAVA using UIMA, and dkpro-tc to perform rapid supervised learning on our dataset. Then we experimented with a large pool of vector-based features for each one of the subtasks.

For the Aspect Term Extraction subtask, the best performing model is obtained by employing Conditional Random Fields (CRF). As features, we utilize the POS-tags of words and their vector representations computed by the word2vec model trained on Yelp data or GoogleNews data. Using these features, we get an F1 score of 78.1 which is substantially better than a baseline of 47.1.

For the Aspect Sentiment Prediction subtask, we obtained the best model by using a one-vs-all SVM model. The best performing model was obtained from using the Rating

Vectors (RV) feature and the Positive/Negative Similarities (PNS) feature. Using these features, we obtain an accuracy score of 72.3 compared to a baseline accuracy of 64.2.

The last subtask we investigated was Aspect Category Detection. In this subtask, we used a multilabel SVM as our model to enable us to assign multiple categories to a single review. The best performing model was obtained from using three features. First, the Normalized Average Vector (NAV) feature, that creates a vector representation for the whole review. Second, the Category Similarities (CS) feature that computes cosine similarities between seed words for each of the possible categories and the words for the reviews. Third, the Token Numbers (TN) feature that counts the number of tokens in the review. For this model we were able to obtain an F1 score of 86.7 compared to a baseline of 65.6.

In summary, we employed vector representations of words to tackle the problem of *Aspect Based Sentiment Analysis*. We introduced several effective vector-based features and showed their utility in addressing the aspect term extraction, aspect category detection, and aspect sentiment prediction sub-tasks. Our vector space approach using these features performed well compared to the baselines, which supports our hypothesis of our ability to capture semantic and sentiment information in the vector space.

## 6.2 Future Work

The purpose of this research is to explore the effectiveness of using vector-based features to perform Aspect Based Sentiment Analysis. In this Chapter, we will discuss some of the ideas for extending the experiments, and extending the research problem.

### 6.2.1 Extending the Experiments

In this subsection, we will discuss some ideas on extending the experiments to improve the performance, mainly by extending the features, and using different datasets to test the robustness and domain dependence of the proposed method.

#### 6.2.1.1 Text-based Features

This thesis was focused on investigating the performance of vector-based features in creating models for Aspect Based Sentiment Analysis. All of the features used, except for POS-tags, were vector based features.

Previous work in this field used text-based features to build ABSA models (Chernyshevich, 2014, Patra et al., 2014, Zhiqiang and Wenting, 2014) and produced good results. The next logical step for our work would be to combine word based features and vector based features together and we expect an increase in the performance.

Some of the text-based features commonly used in sentiment analysis and ABSA:

- POS Frequency: observed aspect terms surrounded by noun or adjectives are also aspects.
- Token Frequency
- Named Entity Feature: whether the token is a named entity
- Head Word
- Head Word POS
- Dependency Relation
- WordNet Taxonomy
- SAO (Subject-Action-Object) Features
- Before the Verb: nouns before the “be” verbs are usually aspects
- Sentiment Words: sentiWordNet collection

For more information about any of the previous features and how they were used, you can refer to the following papers (Chernyshevich, 2014, Patra et al., 2014, Zhiqiang and Wenting, 2014). As mentioned earlier, we expect an improved performance upon including the text-based features with the vector based features.

#### 6.2.1.2 Extra Data Sources

In our experiments, we attempted to apply our ABSA model to automatically annotate the restaurant data collected and annotated by Ganu et al. (2009a). However, we can investigate our model on different datasets in the same domain (e.g. restaurant data obtained from [diningindoha.com](http://diningindoha.com)) or in a different domain (e.g. data on laptop reviews). In fact, it would be interesting to investigate the generalization power of the model by doing cross-dataset training and testing. For example, we could train the model on a Laptop review dataset and test it on a restaurant review dataset for example, or vice

versa. We can also see the effect of training the model on both datasets and testing it on each one of them.

See Appendix A for some of the data sources collected in the line of this research that can be helpful in extending the data sources used.

### 6.2.2 Extending the Research Problem

In this thesis we presented the Aspect Based Sentiment Analysis task as a collection of three subtasks: (1) Aspect Term Extraction, (2) Aspect Sentiment Prediction, (3) Aspect Category Detection (each described in Chapter 3). We have a number of ideas for extending the pool of subtasks to further increase the power of ABSA in analyzing large amounts of user-generated data.

#### 6.2.2.1 Category Sentiment Prediction

One idea is to add another subtask to the pipeline of tasks in ABSA. After extracting the aspect terms, determining their sentiments, and detecting the categories discussed in the review, it would be a good idea to predict the sentiment of each one of the categories detected. We can define the subtask as follows:

Given a set of identified aspect categories for a specific review, we need to classify the sentiment of each category. Each identified category can have one of the following classifications, *positive*, *negative*, *neutral*, or *conflict*. For example, **E1** has negative sentiment with respect to its aspect category “price”, and **E2** have the negative and positive sentiments with respect to its aspect categories “price” and “food”, respectively.

**E1:** “The restaurant was too expensive” → {*price*: negative}

**E2:** “The restaurant was expensive, but the menu was great” → {*price*: negative, *food*: positive}

Generating that information about the sentiment of the category can help us built a more accurate model about the general sentiment towards an entity while preserving some granularity, where sentiment on the category level is less specific than the aspect-level sentiment, but it is more informative than the entity-level sentiments obtained by star ratings and similar methods.

### 6.2.2.2 Aspect Sentiment Summarization

Another path that seems very promising to explore is working with aggregate data on a specific entity. In this thesis, we developed methods to predict aspect-level information for a single review regarding a known entity. Then, we used the aspect information predicted to get some information about categories discussed in each review.

The next step would be to aggregate all the information obtained for the reviews on a specific entity. That is, the aspect-level information for all reviews can be combined to create a coarse model for the aspects at entity level in addition to the review level.

For example, when we analyze restaurant reviews we will have multiple reviews written about a single restaurant. Each review can be analyzed to extract the aspects discussed and their sentiments. Then, we can develop a summarization platform that takes the analysis output for all the reviews and present us with the summary. If many users share the same sentiment about a specific aspect, then that sentiment is more likely to reflect the quality in real life. If the users have different sentiments over the aspect, we can mark that aspect's sentiment as a conflict on the restaurant level.

Having such summarization platform enables us to increase the quality and accuracy of our inference and enables us to achieve some of the applications discussed in the *Motivation* section in Chapter 1.



## Appendix A

# Extra Data Sources

This section will list the different data sources we collected throughout our work on this thesis. Note that most this data was not used in the ABSA application, but is still relevant to sentiment analysis and restaurant reviews.

### A.1 Dining In Doha

The website [diningindoha.com](http://diningindoha.com) is a online review system for restaurants in Doha city in Qatar. Each restaurant has a dedicated page on the website with all the relevant information which made simple to systematically collection data for their website. For the data collection process, we developed Python code using the Scrapy library for web crawling.

Every restaurant's page has the following main sections (page screenshot in Fig A.1):

**About the Restaurant** For every restaurant, the website owners write a paragraph describing the restaurant. This usually includes information about the type of food they serve and the ambiance of the place. These descriptions tend to be very positive.

**Restaurant Details** They also have a section for restaurant details. This section differs from restaurant to another, but many attributes are maintained across restaurants. This includes information about hours, website, parking, phone, and coordinates (viewed in GoogleMaps plugin).

**Our rating** This section includes the rating diningInDoha staff assigned to this restaurant. They rate the following four dimensions (all ratings are out of 5 stars):

1. Food
2. Environment
3. Service
4. Total

**User Rating** DiningInDoha also allows users to leave their rating on the same dimensions mentioned above. This section averages all users ratings.

**Reviews/Comments** The last piece of information included in DiningInDoha's restaurant entry is user comments and reviews. Registered users can leave a comment about the restaurant. Many restaurants have 0-1 comments, but a good number has more than 15 comments.

### A.1.1 Data Collection

As mentioned above, every restaurant has a dedicated page on the website. To collect all the restaurant data, we crawled through every page on the website and organized the data in a JSON format.

All restaraunt information was collected in a single JSON file that is a list of objects, where each JSON object in the list is a restaurant. The structure of a single restaurant object is as follows:

```

{
    Name: __String
    Type: __String
    Address: __String
    Description: __String
    Times: __String
    Website: __String
    Email: __String
    Delivery: __String
    Take_out: __String
    Licensed: __String
    Booking: __String
    Parking: __String
    Phone: __String
    Location: __String
    Reviews: [[username, dateString, reviewText], ...]
}

```

LISTING 1: Dining In Doha information structure in the generated JSON file. The file name is data with reviews.json

The following table (Table A.1) presents a summary of the data that was collected on February 13, 2014.

Source	diningindoha.com
number of restaurants	389
number of descriptions	389
number of reviews	688

TABLE A.1: Summary of the data collected from diningindoha.com on February 13, 2014

## A.2 Labeling Reviews on Amazon Mechanical Turk

One application of the data collected from Dining In Doha is to build a labeled corpus of positive/negative phrases in a restaurant review. We launched an AMT task to have the AMT workers provide this labeling.

The task description was very simple, the user is presented with a restaurant review and is asked to label the phrases that are positive and negative, and he assumes that

**About The Restaurant**

Overlooking the shimmering Arabian Gulf is a new direction for Gordon Ramsay and one of the very best restaurants in Doha. Offering everything which defines the Gordon Ramsay experience in a relaxed, social dining environment, Opal is ideal for families and alike.

At the heart of this bistro-style space are the two live gourmet pizza stations. Complementing the menu's emphasis on spontaneity is a lively wine list, featuring the largest selection in Doha, served by the bottle or the glass, along with the exciting new flavors and textures of fusion and molecular cocktails.

Curious guests are invited to take a drink at The Conservatory which links Opal with Gordon Ramsay, its brother restaurant, to watch the theatre and satisfy a thirst for the out of the ordinary.

**Restaurant details**

<b>Address</b>	
<b>Opening Times</b>	Lunch: 12:00 - 15:00 Dinner: 18:00- 23:00 (Weekdays) 18:00 - 00:00 (Weekends)
<b>Website</b>	<a href="http://www.stregisdoha.com">www.stregisdoha.com</a>
<b>Home Delivery</b>	No
<b>Take Out</b>	No
<b>Licensed</b>	Yes
<b>Booking Recommended</b>	yes
<b>Parking</b>	yes
<b>Telephone</b>	+974 4446 0105

View [St. Regis Hotel Doha](#) in a larger map

**Our rating**

Food	★★★★★
Environment	★★★★★
Service	★★★★★
Total	★★★★★

**User rating**

Food	★★★★★
Environment	★★★★★
Service	★★★★★
Total	★★★★★

By [korsuqi](#) 2nd November 2012

love this place. I love their prawn starter with lime or something and the pizzas are great- I had a greek one with feta cheese on it a few months ago and it was one of the best I've ever had! Its reasonably priced as well you don't go away feeling you been robbed like you do at a lot of places in doha!

**Add your comment**

FIGURE A.1: This figure is a screenshot of what a restaurant page on [diningindoha.com](http://diningindoha.com) looks like.

unlabeled phrases are neutral. Figure A.2, shows a screenshot of the labeling tasks as the workers use it.

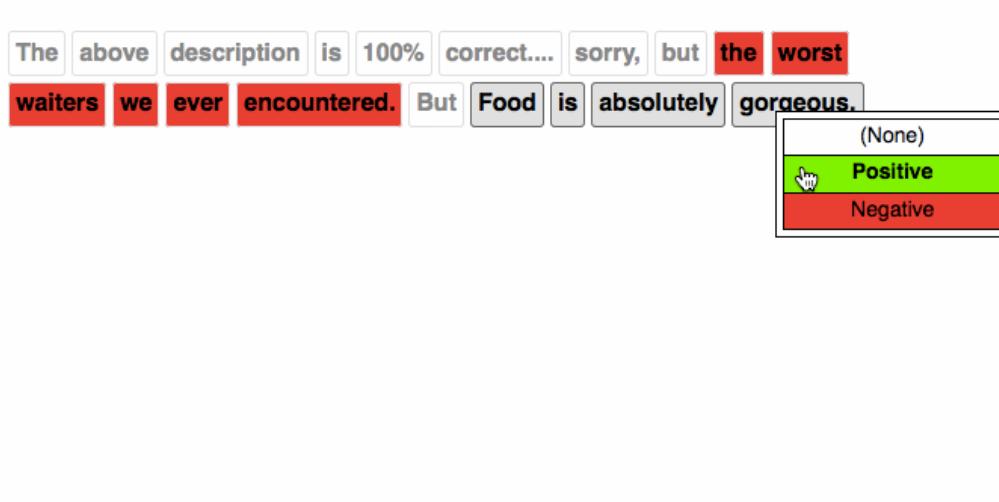


FIGURE A.2: Screenshot of the Amazon Mechanical Turk labeling task. The worker highlights the phrase and indicates whether it is positive or negative. Unlabeled text is assumed to be neutral.

Using this AMT task, we asked workers to label the data we collected from Dining In Doha and built a corpus of annotated reviews.



# Bibliography

- R. Al-Rfou, B. Perozzi, and S. Skiena. Polyglot: Distributed word representations for multilingual NLP. *CoRR*, abs/1307.1662, 2013. URL <http://arxiv.org/abs/1307.1662>.
- Y. Altun, I. Tschantaridis, and T. Hofmann. Hidden Markov support vector machines, 2003a.
- Y. Altun, I. Tschantaridis, T. Hofmann, et al. Hidden Markov support vector machines. In *ICML*, volume 3, pages 3–10, 2003b.
- H. Amiri and T.-S. Chua. Mining slang and urban opinion words and phrases from cqa services: An optimization approach. In *WSDM*, pages 193–202, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-0747-5. doi: 10.1145/2124295.2124319.
- S. Baccianella, A. Esuli, and F. Sebastiani. Multi-facet rating of product reviews. In *ECIR*, pages 461–472, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-00957-0. doi: 10.1007/978-3-642-00958-7\_41.
- M. Bansal, K. Gimpel, and K. Livescu. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 809–815, 2014. URL <http://aclweb.org/anthology/P/P14/P14-2131.pdf>.
- Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.
- P. Blinov and E. Kotelnikov. Blinov: Distributed representations of words for aspect-based sentiment analysis at SemEval 2014. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 140–144. Association for Computational Linguistics, 2014. URL <http://aclweb.org/anthology/S14-2020>.
- J. Bollen, H. Mao, and X. Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8, 2011.

- S. R. Bowman, C. Potts, and C. D. Manning. Recursive neural networks for learning logical semantics. *CoRR*, abs/1406.1827, 2014. URL <http://arxiv.org/abs/1406.1827>.
- J. L. Boyd-Graber, B. Satinoff, H. He, and H. D. III. Besting the quiz master: Crowd-sourcing incremental classification games. In *EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 1290–1301, 2012. URL <http://www.aclweb.org/anthology/D12-1118>.
- 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*, pages 804–812. Association for Computational Linguistics, 2010.
- C. Brun, D. N. Popa, and C. Roux. Xrce: Hybrid classification for aspect-based sentiment analysis. *SemEval 2014*, page 838, 2014.
- Y. Chen and A. I. Rudnicky. Dynamically supporting unexplored domains in conversational interactions by enriching semantics with neural word embeddings. In *Proceedings of the 2014 Spoken Language Technology Workshop, December 7-10, 2014, South Lake Tahoe, Nevada, USA*, pages 590–595, 2014.
- M. Chernyshevich. IHS R&D Belarus: Cross-domain extraction of product features using conditional random fields. *SemEval 2014*, page 309, 2014.
- Y. Choi and C. Cardie. Hierarchical sequential learning for extracting opinions and their attributes. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort ’10, pages 269–274, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL <http://dl.acm.org/citation.cfm?id=1858842.1858892>.
- T. Cohn and P. Blunsom. Semantic role labelling with tree conditional random fields. In *In Proceedings of CoNLL-2005*, pages 169–172, 2005.
- R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.
- J. Daxenberger, O. Ferschke, I. Gurevych, and T. Zesch. Dkpro tc: A Java-based framework for supervised learning experiments on textual data. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 61–66, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL <http://www.aclweb.org/anthology/P14-5011>.

- R. E. de Castilho and I. Gurevych. A broad-coverage collection of portable NLP components for building shareable analysis pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT (OIAF4HLT) at COLING*, pages 1–11, 2014.
- M.-C. de Marneffe and C. D. Manning. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, CrossParser ’08, pages 1–8, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-50-7. URL <http://dl.acm.org/citation.cfm?id=1608858.1608859>.
- M.-C. de Marneffe, C. D. Manning, and C. Potts. “Was It Good? It Was Provocative.” Learning the Meaning of Scalar Adjectives. In *ACL*, pages 167–176, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.
- L. L. P. Doshi. *Using sentiment and social network analyses to predict opening-movie box-office success*. PhD thesis, Massachusetts Institute of Technology, 2010.
- J. Firth. A synopsis of linguistic theory, 1930-1955. *Studies in Linguistic Analysis*, pages 1–32, 1957.
- M. Gamon, A. Aue, S. Corston-Oliver, and E. Ringger. Pulse: Mining customer opinions from free text. In *Advances in Intelligent Data Analysis VI*, pages 121–132. Springer, 2005.
- G. Ganu, N. Elhadad, and A. Marian. Beyond the stars: Improving rating predictions using review text content, 2009a.
- G. Ganu, N. Elhadad, and A. Marian. Beyond the stars: Improving rating predictions using review text content. In *WebDB*, volume 9, pages 1–6, 2009b.
- D. Z. Guo, G. Tur, W. Yih, and G. Zweig. Joint semantic utterance classification and slot filling with recursive neural networks. pages 554–559, 2014.
- Y. Hong and S. Skiena. The wisdom of bookies. sentiment analysis versus the NFL point spread. 2010.
- 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*, pages 168–177. ACM, 2004.
- M. Iyyer, J. L. Boyd-Graber, L. M. B. Claudino, R. Socher, and H. D. III. A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the*

- ACL*, pages 633–644, 2014. URL <http://aclweb.org/anthology/D/D14/D14-1070.pdf>.
- Y. Jo and A. H. Oh. Aspect and sentiment unification model for online review analysis. In *WSDM*, pages 815–824, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0493-1. doi: 10.1145/1935826.1935932. URL <http://doi.acm.org/10.1145/1935826.1935932>.
- T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009.
- J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1. URL <http://dl.acm.org/citation.cfm?id=645530.655813>.
- O. Levy and Y. Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 302–308, 2014. URL <http://aclweb.org/anthology/P/P14/P14-2050.pdf>.
- C. Lin and Y. He. Joint sentiment/topic model for sentiment analysis. In *CIKM*, pages 375–384, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-512-3. doi: 10.1145/1645953.1646003. URL <http://doi.acm.org/10.1145/1645953.1646003>.
- B. Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012.
- Y. Liu, X. Huang, A. An, and X. Yu. Arsa: a sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 607–614. ACM, 2007.
- C. Long, J. Zhang, and X. Zhut. A review selection approach for accurate feature rating estimation. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 766–774. Association for Computational Linguistics, 2010.
- B. Lu, M. Ott, C. Cardie, and B. K. Tsou. Multi-aspect sentiment analysis with topic models. In *ICDMW*, pages 81–88, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-0-7695-4409-0. doi: 10.1109/ICDMW.2011.125.
- C. Manning, T. Grow, T. Grenager, J. Finkel, and J. Bauer. *StanfordTokenizer*. <http://nlp.stanford.edu/software/tokenizer.shtml>, 2010.

- Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai. Topic sentiment mixture: Modeling facets and opinions in weblogs. In *WWW*, pages 171–180, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-654-7. doi: 10.1145/1242572.1242596. URL <http://doi.acm.org/10.1145/1242572.1242596>.
- A. Mejer and K. Crammer. Confidence in structured-prediction using confidence-weighted models. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 971–981. Association for Computational Linguistics, 2010.
- T. Mikolov. *Statistical Language Models Based on Neural Networks*. PhD thesis, Brno University of Technology, 2012.
- T. Mikolov. Word2Vec. <https://code.google.com/p/word2vec/>, 2014.
- T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013a. URL <http://arxiv.org/abs/1301.3781>.
- T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013b. URL <http://arxiv.org/abs/1310.4546>.
- T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013c. URL <http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf>.
- T. Mikolov, W. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In *NAACL*, pages 746–751, 2013d. URL <http://aclweb.org/anthology/N/N13/N13-1090.pdf>.
- J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.
- B. O’Connor, R. Balasubramanyan, B. R. Routledge, and N. A. Smith. From Tweets to polls: Linking text sentiment to public opinion time series. *ICWSM*, 11:122–129, 2010.
- P. Ogren and S. Bethard. Building test suites for UIMA components. In *Proceedings of the Workshop on Software Engineering, Testing, and Quality Assurance for*

- Natural Language Processing (SETQA-NLP 2009)*, pages 1–4, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL <http://www.aclweb.org/anthology/W/W09/W09-1501>.
- N. Okazaki. Crfsuite: a fast implementation of conditional random fields (CRFs), 2007. URL <http://www.chokkan.org/software/crfsuite/>.
- B. Pang and L. Lee. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *ACL*, pages 115–124, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219855. URL <http://dx.doi.org/10.3115/1219840.1219855>.
- B. Pang and L. Lee. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, Jan. 2008. ISSN 1554-0669. doi: 10.1561/1500000011. URL <http://dx.doi.org/10.1561/1500000011>.
- B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: Sentiment classification using machine learning techniques. In *EMNLP*, pages 79–86, Stroudsburg, PA, USA, 2002. doi: 10.3115/1118693.1118704. URL <http://dx.doi.org/10.3115/1118693.1118704>.
- B. G. Patra, S. Mandal, D. Das, and S. Bandyopadhyay. Ju\_cse: A conditional random field (CRF) based approach to aspect based sentiment analysis. *SemEval 2014*, page 370, 2014.
- M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar. SemEval-2014 task 4: Aspect based sentiment analysis. In *SemEval*, pages 27–35. Association for Computational Linguistics, 2014. URL <http://aclweb.org/anthology/S14-2004>.
- A.-M. Popescu and O. Etzioni. Extracting product features and opinions from reviews. In *Natural language processing and text mining*, pages 9–28. Springer, 2007.
- M. Sahlgren. *The Word-Space Model*. PhD thesis, Stockholm University, 2006.
- H. Schütze. Dimensions of meaning. In *Proceedings of Supercomputing '92*, pages 787–796, 1992a.
- H. Schütze. Word space. In *NIPS*, pages 895–902, 1992b.
- F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 134–141. Association for Computational Linguistics, 2003.

- S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.
- R. Socher. *Recursive Deep Learning for Natural Language Processing and Computer Vision*. PhD thesis, Stanford University, 2014.
- 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)*, volume 1631, page 1642. Citeseer, 2013.
- T. T. Thet, J.-C. Na, and C. S. Khoo. Aspect-based sentiment analysis of movie reviews on discussion boards. *Journal of Information Science*, page 0165551510388123, 2010.
- K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL-HLT*, pages 173–180, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1073445.1073478.
- P. D. Turney. Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 417–424. Association for Computational Linguistics, 2002.
- J. Wagner, P. Arora, S. Cortes, U. Barman, D. Bogdanova, J. Foster, and L. Tounsi. Dcu: Aspect-based polarity classification for SemEval task 4. 2014.
- L. Xu, K. Liu, S. Lai, Y. Chen, and J. Zhao. Mining opinion words and opinion targets in a two-stage framework. In *ACL, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 1764–1773. The Association for Computer Linguistics, 2013. URL <http://aclweb.org/anthology/P/P13/P13-1173.pdf>.
- B. Yang and C. Cardie. Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1640–1649, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL <http://www.aclweb.org/anthology/P13-1161>.
- W. Zhang and S. Skiena. Trading strategies to exploit blog and news sentiment. In *ICWSM*, 2010.
- 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*, EMNLP ’10, pages 56–65, Stroudsburg, PA,

- USA, 2010. Association for Computational Linguistics. URL <http://dl.acm.org/citation.cfm?id=1870658.1870664>.
- Y. Zhao, B. Qin, and T. Liu. Collocation polarity disambiguation using web-based pseudo contexts. In *EMNLP-CoNLL*, pages 160–170, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL <http://dl.acm.org/citation.cfm?id=2390948.2390968>.
- T. Zhiqiang and W. Wenting. Dlirec: Aspect term extraction and term polarity classification system. 2014.
- X. Zhou, X. Wan, and J. Xiao. Collective opinion target extraction in Chinese microblogs. In *EMNLP*, pages 1840–1850, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL <http://www.aclweb.org/anthology/D13-1189>.
- X. Zhu, S. Kiritchenko, and S. M. Mohammad. Nrc-canada-2014: Recent improvements in the sentiment analysis of tweets. *SemEval 2014*, page 443, 2014.