Adaptation of Sentiment Analysis to New Linguistic Features, Informal Language Form and World Knowledge

M.Tech. Dissertation

Submitted in partial fulfillment of the requirements of the degree of Master of Technology

by

Subhabrata Mukherjee

Roll No: 10305061

Supervisor: Dr. Pushpak Bhattacharyya

July 1, 2012

Indian Institute of Technology, Bombay

Department of Computer Science and Engineering

Dissertation Approval Certificate

Department of Computer Science and Engineering

Indian Institute of Technology, Bombay

The dissertation entitled **"Sentiment Analysis: A Multi-Dimensional Perspective"**, submitted by **Subhabrata Mukherjee** (Roll No: **10305061**) is approved for the degree of **Master of Technology** in **Computer Science and Engineering** from **Indian Institute of Technology, Bombay**.

Prof. Pushpak Bhattacharyya
CSE Department, IIT Bombay
Supervisor

Prof. Ganesh Ramakrishnan
CSE Department, IIT Bombay
Internal Examiner

Mr. Girish Palshikar
Principal Scientist, TCS, Pune
External Examiner

Prof. Malhar Kulkarni
HSS Department, IIT Bombay
Chairperson

Place: IIT Bombay, Mumbai
Date: 12^{th} June, 2012

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

Subhabrata Mukherjee

Roll Number: 10305061

Date: 1/7/2012

IIT Bombay

Acknowledgements

I would like to thank my guide Dr. Pushpak Bhattacharyya for his valuable feedback and suggestions time to time. I am thankful to Balamurali A.R. for the critical reviews of all of my works. I am thankful to Aditya Joshi and Akshat Malu who have helped me during certain phases of this work. I am also thankful to Ashutosh Kumar Nirala, Janardhan Singh, Srijit Dutt for various discussions related to my work.

Abstract

In this work, we investigate the adaptation of sentiment analysis to *linguistic features* (in the form of *dicourse coherency* and *feature specificity), informal text form* (posts on *Twitter* called *tweets*) and *world knowledge*. We study opinion in the social media through *three* systems – *TwiSent* (Twitter), *WikiSent* (Wikipedia) and *YouCat* (Youtube).

In order to incorporate *feature specificity*, we analyze the association between words, forming an opinion expression about a specific feature, using *dependency parsing*. As an approach to incorporate *discourse coherency*, we propose a lightweight method of incorporating *discourse information,* in a *bag-of-words* model, that proves beneficial in resource-constrained scenarios. In the context of understanding opinion in social media content, we present a multi-stage system, *TwiSent*, for sentiment analysis in *Twitter*. It addresses the issues of *spam*, *noisy text*, *pragmatics* and *entity specificity* in Twitter. Finally, we present an approach to incorporate world knowledge in a sentiment analysis system through WikiSent, which harvests information from *Wikipedia* to create a *topic-specific*, *extractive summary* of a review. Motivated by the effectiveness of concept expansion approach in *WikiSent* and the social media content in *TwiSent*, we propose an unsupervised system, *YouCat*, which combines both the approaches for video genre prediction, from meta-data and user comments of a video. Prompted by the requirement of a metric for feature replacement in the sentiment analysis of reviews, we introduce *SenSim* as a *sentiment-semantic similarity metric*. We perform a number of experiments, in the *movie review* domain, *travel review* domain and the micro-blog *Twitter*, to validate our claims. We achieve better accuracies than the state-of-the-art systems in many of these experiments.

TABLE OF CONTENTS

Table of Figures

Chapter 1

1. INTRODUCTION

1.1 WHAT IS SENTIMENT ANALYSIS?

Sentiment Analysis is a *Natural Language Processing* and *Information Extraction* task that aims to obtain writer's feelings expressed in positive or negative comments, questions and requests, by analyzing a large numbers of documents. Generally speaking, sentiment analysis aims to determine the attitude of a speaker or a writer with respect to some topic or the overall tonality of a document. In recent years, the exponential increase in the Internet usage and exchange of public opinion is the driving force behind Sentiment Analysis today. The Web is a huge repository of structured and unstructured data. The analysis of this data to extract latent public opinion and sentiment is a challenging task.

> Liu *et al.* (2009) defines a sentiment or opinion as a quintuple-
>
> *"$<o_j, f_{jk}, so_{ijkl}, h_i, t_l>$, where o_j is a target object, f_{jk} is a feature of the object o_j, so_{ijkl} is the sentiment value of the opinion of the opinion holder h_i on feature f_{jk} of object o_j at time t_l, so_{ijkl} is +ve,-ve, or neutral, or a more granular rating, h_i is an opinion holder, t_l is the time when the opinion is expressed."*

The analysis of sentiments may be document based where the sentiment in the entire document is summarized as positive, negative or objective. It can be sentence based where individual sentences, bearing sentiments, in the text are classified. SA can be phrase based where the phrases in a sentence are classified according to polarity.

Sentiment Analysis identifies the phrases in a text that bears some sentiment. The author may speak about some *objective facts* or *subjective opinions*. It is necessary to distinguish between the two. SA finds the subject towards whom the sentiment is directed. A text may contain many entities but it is necessary to find the entity towards which the sentiment is directed. It identifies the polarity and degree of the sentiment. Sentiments are classified as *objective* (facts), *positive* (denotes a state of happiness, bliss or satisfaction on part of the writer) or *negative* (denotes a state of sorrow, dejection or disappointment on part of the writer). The sentiments can further be given a score based on their *degree* of positivity, negativity or objectivity.

1.2 APPLICATIONS OF SENTIMENT ANALYSIS

Word of mouth (WOM) is the process of conveying information from person to person and plays a major role in customer buying decisions. In commercial situations, WOM involves consumers sharing attitudes, opinions, or reactions about businesses, products, or services with other people. WOM communication functions based on social networking and trust. People rely on families, friends, and others in their social network. Research also indicates that people appear to trust seemingly disinterested opinions from people outside their immediate social network, such as online reviews. This is where Sentiment Analysis comes into play. Growing availability of opinion rich resources like online review sites, blogs, social networking sites have made this "decision-making process" easier for us. With explosion of Web 2.0 platforms consumers have a soapbox of unprecedented reach and power by which they can share opinions. Major companies have realized these consumer voices affect shaping voices of other consumers.

Sentiment Analysis thus finds its use in *Consumer Market* for Product reviews, *Marketing* for knowing consumer attitudes and trends, *Social Media* for finding general opinion about recent hot topics in town, *Movie* to find whether a recently released movie is a hit.

Pang-Lee *et al.* (2002) broadly classifies the applications into the following categories.

a. Applications to Review-Related Websites

   Movie Reviews, Product Reviews *etc.*

b. Applications as a Sub-Component Technology

   Detecting antagonistic, heated language in mails, spam detection, context sensitive information detection *etc.*

c. Applications in Business and Government Intelligence

   Knowing Consumer attitudes and trends

d. Applications across Different Domains

   Knowing public opinions for political leaders or their notions about rules and regulations in place *etc.*

1.3 CHALLENGES FOR SENTIMENT ANALYSIS

Sentiment Analysis approaches aim to extract *positive* and *negative* sentiment bearing words from a text and classify the text as *positive, negative* or else *objective* if it cannot find any sentiment bearing words. In this respect, it can be thought of as a text categorization task. In text classification there are many classes corresponding to different topics whereas in Sentiment Analysis we have only 3 broad classes. Thus it seems Sentiment Analysis is easier than text classification which is not quite the case. The general challenges can be summarized as:

1.3.1 Implicit Sentiment and Sarcasm

A sentence may have an implicit sentiment even without the presence of any sentiment bearing words. Consider the following examples.

How can anyone sit through this movie?

One should question the stability of mind of the writer who wrote this book.

Both the above sentences do not explicitly carry any negative sentiment bearing words although both are negative sentences. Thus *identifying semantics* is more important in SA than *syntax detection.*

1.3.2 Domain Dependency

There are many words whose polarity changes from domain to domain. Consider the following examples.

The story was unpredictable.

The steering of the car is unpredictable.

Go read the book.

In the first example, the sentiment conveyed is positive whereas the sentiment conveyed in the second is negative. The third example has a positive sentiment in the book domain but a negative sentiment in the movie domain (where the director is being asked to go and read the book).

1.3.3 Thwarted Expectations

Sometimes the author deliberately sets up context only to refute it at the end. Consider the following example:

This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it can't hold up.

Inspite of the presence of words that are positive in orientation the overall sentiment is negative because of the crucial last sentence, whereas in traditional text classification this would have been classified as positive as *term frequency* is more important there than *term presence.*

1.3.4 Pragmatics

It is important to detect the pragmatics of user opinion which may change the sentiment thoroughly. Consider the following examples:

I just finished watching Barca DESTROY Ac Milan

That final completely destroyed me.

Capitalization can be used with subtlety to denote sentiment. The first example denotes a positive sentiment whereas the second denotes a negative sentiment. There are many other ways of expressing pragmatism.

1.3.5 World Knowledge

Often world knowledge needs to be incorporated in the system for detecting sentiments. Consider the following examples:

He is a Frankenstein.

Just finished Doctor Zhivago for the first time and all I can say is Russia sucks.

The first sentence depicts a negative sentiment whereas the second one depicts a positive sentiment. But one has to know about *Frankenstein and Doctor Zhivago* to find out the sentiment.

1.3.6 Subjectivity Detection

This is to differentiate between opinionated and non-opinionated text. This is used to enhance the performance of the system by including a subjectivity detection module to filter out objective facts. But this is often difficult to do. Consider the following examples:

I hate love stories.

I do not like the movie "I hate stories".

The first example presents an objective fact whereas the second example depicts the opinion about a particular movie.

1.3.7 Entity Identification

A text or sentence may have multiple entities. It is extremely important to find out the entity towards which the opinion is directed. Consider the following examples.

Samsung is better than Nokia

Ram defeated Hari in football.

The examples are positive for Samsung and Ram respectively but negative for Nokia and Hari.

1.3.8 Negation

Handling negation is a challenging task in SA. Negation can be expressed in subtle ways even without the explicit use of any negative word. A method often followed in handling negation explicitly in sentences like "*I do not like the movie",* is to reverse the polarity of all the words appearing after the negation operator (like *not)*. But this does not work for "*I do not like the acting but I like the direction".* So we need to consider the *scope* of negation as well, which extends only till *but* here. So the thing that can be done is to change polarity of all words appearing after a negation word till another negation word appears. But still there can be problems. For example, in the sentence "*Not only did I like the acting, but also the direction",* the polarity is *not reversed* after "not" due to the presence of "*only".* So this type of combinations of "not" with other words like "only" has to be kept in mind while designing the algorithm.

1.4 MOTIVATION FOR CURRENT WORK

1.4.1 Sentiment Analysis in Twitter with Lightweight Discourse Analysis

An essential phenomenon in natural language processing is the use of discourse relations to establish a coherent relation, linking phrases and clauses in a text.

The presence of linguistic constructs like *connectives, modals, conditionals* and *negation* can alter sentiment at the sentence level as well as the clausal or phrasal level. Consider the example, "*@user share 'em! i'm* **quite excited** *about Tintin, despite* **not really liking** *original comics. Probably because Joe Cornish had a hand in."* The overall sentiment of this example is *positive*, although there is equal number of positive and negative words. This is due to the connective *despite* which gives more weight to the previous discourse segment. Any bag-of-words model would be unable to classify this sentence without considering the discourse marker. Consider another example, "*Think i'll stay with the whole 'sci-fi'* **shit***. but this time...a* **classic** *movie."* The overall sentiment is again *positive* due to the connective *but,* which gives more weight to the following discourse segment. Thus it is of utmost importance to capture all these phenomena in a computational model.

Traditional works in *discourse analysis* use a discourse parser trained on Penn-Discourse-Treebank (Marcu 2000; Zirn *et al.*, 2011, Wellner *et al.*; 2006; Pitler *et al.*, 2009; Elwell *et al.*, 2008) or a dependency parser (Vincent *et al.*, 2006). Many of these works and some other works in discourse (Taboada *et al.*, 2008; Zhou et al., 2011) build on the Rhetorical Structure Theory (RTS) proposed by Mann *et al.* (1988) which tries to identify the relations between the nucleus and satellite in the sentence.

Most of these theories are well-founded for *structured text,* and *structured* discourse annotated corpora are available to train the models. However, using these methods for micro-blog discourse analysis pose some fundamental difficulties:

1. Micro-blogs, like *Twitter*, do not have any restriction on the form and content of the user posts. Users do not use formal language to communicate in the micro-blogs. As a result, there are abundant *spelling mistakes, abbreviations, slangs, discontinuities* and *grammatical errors*. This can be observed in the given examples from real-life *tweets*. The errors cause natural language processing tools like *parsers* and *taggers* to fail frequently (Dey *et al.*, 2009). As the tools are generally trained on structured text, they are unable to handle the noisy and unstructured text in this medium. Hence most of the discourse-based

methods, based on RST or parsing of some form, will be unable to perform very well in micro-blog data.

2. The web-based applications require a fast response time. Using a heavy linguistic resource, like *parsing,* increases the processing time and slows down the application. Thus *parsing* is not commonly used for real-time interactive systems.

Most of the works in micro-blogs, like *Twitter,* (Alec *et al.*, 2009; Read *et al.*, 2005; Pak *et al.*, 2010; Gonzalez *et al.*, 2011) use a bag-of-words model with features like part-of-speech information, unigrams, bigrams *etc.* along with other domain-specific, specialized features like *emoticons, hashtags etc.* In *most* of these works, the *connectives, modals* and *conditionals* are simply ignored as stop words during feature vector creation. Hence, the discourse information that can be harnessed from these elements is completely discarded. In this work, we show how the *connectives, modals, conditionals* and *negation* based discourse information can be incorporated in a bag-of-words model to give better sentiment classification accuracy.

1.4.2 Feature Specific Sentiment Analysis for Product Reviews

The sentiment regarding a particular product in a review is seldom explicitly positive or negative; rather people tend to have a mixed opinion about various features, some positive and some negative. Thus the feature specific opinion matters more than the overall opinion.

Consider a review "*I like Micromax's multimedia features but the battery life sucks."* This sentence has a mixed emotion. The emotion regarding *multimedia* is positive whereas that regarding *battery life* is negative. Hence, it is of utmost importance to extract only those opinions relevant to a particular feature (like *battery life or multimedia*) and classify them, instead of taking the complete sentence and the overall sentiment.

In this work, we propose a method that represents the features and corresponding opinions in the form of a graph where we use dependency parsing to capture the relations between the features and their associated opinions. The idea is to capture the association between any specific feature and the expressions of opinion that come together to describe that feature. This is done by capturing the *short range* and *long range dependencies* between the words using dependency parsing.

1.4.3 TwiSent: A Multi-Stage System for Sentiment Analysis in Twitter

Social media sites, like Twitter, generate voluminous amounts of data. At present, around 250 million tweets are generated daily[1]. This information content could be leveraged to create applications that have a social as well as an economic value.

In this work, we present a multistage system, *TwiSent*, to analyze the sentiment of tweets based on the topic searched in Twitter. TwiSent retrieves tweets pertaining to the topic searched and categorizes them based on their sentiment content into *positive, negative* and *objective* classes. In addition to this, we generate a sentiment score by aggregating the sentiment of all the tweets. This acts as a sentiment snap shot of the given topic as represented by the web. A text limit of 140 characters per tweet makes Twitter a noisy medium[2] for text analysis tasks. Compared to other text genres like News, Blogs *etc.*, it has a poor syntactic and semantic structure. For example, consider the following tweet "*Had Hella fun today with the team. Y'all are hilarious! &Yes, i do need more black homies.....* ". Apart from the irregular syntax, the following sentence has other problems like *slangs, ellipses, nonstandard vocabulary etc.* A direct analysis of such noisy text using commonly applied Natural Language Processing (NLP) tools would be futile. The problem is compounded by the increasing number of *spams* in Twitter like *promotional* tweets, *bot-generated* tweets, *random links* to other websites *etc*. In fact Twitter contains around 40% tweets as pointless babble[3]. In this work, we tackle the following problems which are exclusive to a micro-blog genre like Twitter for assessing sentiment content.

1. **Twitter based spam:** We introduce the concept of spams in Twitter and define it in the context of sentiment analysis. We model it as a partially supervised classification problem.
2. **Spell checker for noisy text:** Short text length leads to irregular and truncated spellings. We propose a new text normalization method to address this problem.
3. **Entity detection:** The objective of our system is to analyze the sentiment of a tweet based on the topic and not its overall sentiment. For example the tweet, "*Samsung multimedia features are better than that of Nokia",* is positive w.r.t *Samsung* but it is negative w.r.t *Nokia*. A module for entity based sentiment detection is introduced to assess the sentiment of the topic in tweets.

[1] http://techcrunch.com/2011/10/17/Twitter-is-at-250-milliontweets-per-day/
[2] http://www.kantar.com/pdf/social_in_context_part2.pdf
[3] http://cs.wellesley.edu/~cs315/Papers/What%20is%20twitter-a%20social%20net%20or%20news%20media.pdf

4. **Pragmatics detection:** We make an attempt to detect the pragmatics involved in tweets in the form of *emoticons, capitalization, vowel elongation etc.* and leverage them for sentiment detection.

1.4.4 WikiSent: A Weakly Supervised System for Sentiment Analysis using Extractive Summarization with Wikipedia

In the movie domain there has been a flurry of review sites giving critics view about the various aspects of a movie. This is of importance not only to the people directly related to the movie-making but also to the audience, whose viewing decisions are quite influenced by these reviews.

Sentiment analysis of movie reviews aims to automatically infer the opinion of the movie reviewer. It often generates a rating on a pre-defined scale. Automated analysis of movie reviews is quite a challenge in text classification (Turney, 2002) due to the various nuances associated with the critic reviews. The author may talk about a lot of topics which are not directly related to the movie in focus. Tightly intermixed with various objective statements are his subjective opinions about the movie, which are quite difficult to extract. An objective statement is not just a factual statement, but is objective from the point of view of analyzing the opinion about the movie.

This work is different from traditional automatic text summarization or abstractive summarization. This is because the objective is not to obtain a shorter text but to retrieve *relevant opinionated text*. This focused extraction requires external world knowledge about the various technical aspects of the movie (like *movie plot, film crew, characters, domain specific features etc.*). Wikipedia feeds the system with this technical knowledge which is used to create an extract of the review. This extract is subsequently classified by a lexicon.

Consider the fragment of a review of the movie L.I.E taken from the IMDB movie review corpus (Pang *et al.*, 2002) which has been tagged as a negative review:

> [1]Best remembered for his understated performance as Dr. Hannibal Lecter in Michael Mann's forensics thriller, Manhunter, Scottish character actor Brian Cox brings something special to every movie he works on. [2]Usually playing a bit role in some studio schlock (he dies halfway through The Long Kiss Goodnight), he's only occasionally given something meaty and substantial to do. [3]If you want to see some brilliant acting, check out his work as a dogged police inspector opposite Frances McDormand in Ken Loach's

Hidden Agenda.

[4]Cox plays the role of Big John Harrigan in the disturbing new indie flick L.*I.E.*, which Lot 47 picked up at Sundance when other distributors were scared to budge. [5]Big John feels the love that dares not speak its name, but he expresses it through seeking out adolescents and bringing them back to his pad. [6]What bothered some audience members was the presentation of Big John in an oddly empathetic light. [7]He's an even-tempered, funny, robust old man who actually listens to the kids' problems (as opposed to their parents and friends, both caught up in the high-wire act of their own confused lives.). [8]He'll have sex-for-pay with them only after an elaborate courtship, charming them with temptations from the grown-up world"

……

[9]It's typical of unimaginative cinema to wrap things up with a bullet, sparing the writers from actually having to come up with a complex, philosophical note . [10]In this regard, l.*i.e.* (and countless other indie films) share something in common with blockbuster action films : problems are solved when the obstacle is removed . [11]How often does real life work this way? to extend the question : if a movie is striving for realism , do dramatic contrivances destroy the illusion ?"

Example 1.1: Review of the Movie L.I.E

The first paragraph of the review talks about the central character Brian Cox's notable performance in some earlier movie. The second paragraph gives a brief description of his character in an empathetic light which comprises of positive opinions about the character. The reviewer opinion about the movie comes only in the last paragraph, where he gives some negative opinions about the movie. The review consists of majority positive words, not all of which are significant to the reviewer opinion, outweighing the negative opinions about the movie. A bag-of-words classifier, thus, would wrongly classify this review as positive.

We propose a system that harnesses Wikipedia *infobox-class ontological information* (Wu *et al.*, 2008) to incorporate World Knowledge in a system, to obtain an *extractive opinionated summary* of a movie review. This, in turn, helps in sentiment classification of the review due to the filtering out of objective concepts from subjective opinions. This work is *mostly* unsupervised, requiring no labeled training data. The weak supervision comes from the

usage of resources like a POS-tagger (statistically trained) and Sentiment Lexicons (manually constructed or statistically trained), due to their mode of construction. This work also presents an analysis of the movie review dataset (Pang *et al.*, 2002) and the trends persisting there. Although there have been more than 100 works[4] on this dataset, this work is the first one to analyze it from different perspectives like the movie genre, year of release and polarity. Although this work deals only with movie reviews, a similar approach can be used in any domain for sentiment analysis.

1.4.5 YouCat: A Weakly Supervised System for Youtube Video Categorization from User Comments and Meta Data using WordNet and Wikipedia

In recent times there has been an explosion in the number of online videos. With the gradually increasing multimedia content, the task of efficient query-based video retrieval has become important, for which the proper genre or category identification of the video is essential. The automatic genre identification of videos has been traditionally posed as a supervised classification task of the features derived from the audio, visual content and textual features. Whereas some works focus on classifying the video based on the *meta data (text)* provided by the uploader, other works attempt to extract low-level features by analyzing the *frames, signals, audio etc*. along with textual features. There have been some recent advances in incorporating new features for classification like the social content comprising of the *user connectivity*, *comments, interest etc*.

All the above approaches pose the genre prediction task as a supervised classification requiring a large amount of training data. It has been argued that a *serious challenge* for supervised classification is the collection of manually labeled data (Filippova *et al.*, 2010; Wu *et al.*, 2010; Zanetti *et al.*, 2008). For example, consider a video with a descriptor "*It's the NBA's All-Mask Team!*". Unless there is a video in the training set with *NBA* in the video descriptor labeled with *Sport*, there is no way of associating *NBA* to *Sport*. It is also not possible to associate *NBA* to *Basketball* and then to *Sport*. So as *new genre-related concepts* (like new sports, technologies, domain-dependent terms *etc*.) appear every day, the training set should expand incorporating all these new concepts which makes training very expensive. The problem is compounded by the *noisy* and *ambiguous* text prevalent in the media due to the *slangs*, *acronyms etc.*; and essentially *very short text* provided by the user, for title and video description, which provide little information (Wu *et al.,* 2010). The focus of this work is to propose an unsupervised system that requires no labeled data for training and can be easily

[4] http://www.cs.cornell.edu/people/pabo/movie-review-data/otherexperiments.html

extended to identify new categories. The system can easily adapt to the changing times incorporating world knowledge to overcome the labeled data shortage. It extracts all the features from the video uploader provided meta-data like the *video title*, *description of the video* as well as the *user comments*. The system incorporates social content by analyzing the user comments on the video, which is essential as often the meta-data associated with a video is absent or not adequate enough to predict its category. *WordNet* and *Wikipedia* are used as world knowledge sources for *expanding* the video descriptor which is necessary since the uploader provided text is frequently very short, as are the user comments. WordNet is used for knowing the meaning of an unknown word whereas Wikipedia is used for expanding the *named entities* (which are mostly absent in the WordNet) like "*NBA"* in the given example. In this work, we show how the textual features can be analyzed with the help of WordNet and Wikipedia to predict the video category without requiring any labeled training set.

In this work, we propose a completely unsupervised system, *YouCat*, for categorizing Youtube[5] videos into different genres like *Comedy, Horror, Romance, Sports*, *Information Technology etc*. The key aspects of the work can be summarized as: (1) Unlike other genre identification works, which are *mostly* supervised, we present a completely unsupervised system requiring *no labeled data for training* (2) It extracts information from the *video title*, *meta description,* user comments (which together form the *video descriptor*) and uses *Wikipedia*, *WordNet* for concept expansion (3) The system can easily incorporate new genres without any supervision (4) The proposed algorithm with a time complexity of $O(|W|)$ (where (|W|) is the number of words in the video descriptor) is efficient to be deployed in the web for real-time video categorization.

1.4.6 Leveraging Sentiment to Compute Word Similarity

Use of similarity metrics is unavoidable in many Natural Language Processing (NLP) systems like *sense disambiguation* (Banerjee & Pedersen 2002), *malapropism detection* (Hirst & St-Onge 1997), *context sensitive spelling correction* (Patwardhan 2003) *etc*. The underlying principle of these metrics has been distributional similarity in terms of their meaning. For example, *refuge* and *asylum* are similar words because they have the same meaning and similar set of words accompany them in a given context. Based on the meaning alone, these words are mutually replaceable.

[5] www.youtube.com

At present, there are various advanced text editors which have the ability to replace a word based on the meaning suitable for the domain/genre in which article is being written. To select an appropriate replacement word, they follow a similarity based on meaning alone. Motivated by the idea of sub-languages (Grishman 2001), we believe similarity based on meaning alone cannot suffice this need. For example, in the previous case, even though *refuge* can be replaced with *asylum*, *mad house* cannot be used to do so. This is because *mad house* evokes a negative connotation or sentiment which makes the word unsuitable for replacement.

In this work, we introduce a new WordNet based similarity metric, SenSim, which incorporates sentiment content (*i.e.*, degree of positive or negative sentiment) of the words being compared to measure the similarity. The proposed metric is based on the hypothesis that knowing the sentiment is beneficial in measuring the similarity. To verify this hypothesis, we measure and compare the annotator agreement for 2 annotation strategies: 1) sentiment information of a pair of words is considered while annotating and 2) sentiment information of a pair of words is not considered while annotating. Inter-annotator correlation scores show that the agreement is better when the two annotators consider sentiment information while assigning a similarity score to a pair of words.

We create vector representations of Word-Net glosses and compare their cosines to calculate the similarity score. To include the sentiment content of the words being compared, we include sentiment scores of the content words of the gloss into the vector. The main contribution of this work is in addressing the following question:

Does inclusion of sentiment content as an additional parameter for comparison improve similarity measurement?

Roadmap

Chapter 1 gives a brief introduction of sentiment analysis, its applications and challenges followed by the motivation behind our work.

Chapter 2 gives an extensive literature survey of the prominent works in the field of sentiment analysis. Chapter 3 discusses works specifically related to our work.

In *Chapter 4*, we propose a lightweight method for using discourse relations for polarity detection of *tweets*. This method is targeted towards the web-based applications that deal with *noisy, unstructured* text, like the *tweets,* and cannot afford to use heavy linguistic resources like *parsing* due to noisy text and requirement of a fast response time.

In *Chapter 5*, we present a novel approach to identify *feature specific* expressions of opinion, in product reviews with *different features* and *mixed emotions*. The objective is realized by identifying a set of potential features in the review and extracting opinion expressions about those features by exploiting their associations via dependency parsing.

In *Chapter 6*, we present *TwiSent*, a sentiment analysis system for Twitter. Based on the topic searched, TwiSent collects tweets pertaining to it and categorizes them into the different polarity classes *positive, negative* and *objective*. Through *TwiSent*, we address some genuine problems in Micro-blogs like: 1) *Spams* pertaining to sentiment analysis in Twitter, 2) *Structural anomalies in the text* in the form of incorrect spellings, nonstandard abbreviations, slangs *etc*., 3) *Entity specificity* in the context of the topic searched and 4) Pragmatics embedded in the text, though naively.

Chapter 7 describes a *weakly* supervised system, *WikiSent*, for sentiment analysis of movie reviews. The objective is to classify a movie review as *positive* or *negative*, based on those sentences bearing opinion on the movie alone, leaving out other irrelevant text. WikiSent uses world knowledge to create an *extractive opinionated summary* of the review. The filtering out of irrelevant or objective concepts helps in polarity classification of the review. The *ontological* and *sectional* information in Wikipedia, automatically extracted from the infobox-class, is used to incorporate *movie* and *genre-specific* knowledge into the system.

In *Chapter 8*, we propose a weakly supervised system, *YouCat*, which uses the concept expansion approach in *WikiSent* and incorporates the social media content, as in *TwiSent*. It extracts information from the *video title*, *meta description,* user comments and uses *Wikipedia*, *WordNet* for concept expansion. It uses an overlap-based approach for genre

classification based on a set of cue words constructed using a manually specified, small seed list of words (~ 1-3 words) and a thesaurus.

In *Chapter 9*, we introduce a new WordNet based similarity metric, *SenSim*, which incorporates sentiment content of the words being compared to measure the similarity between them. This is based on the hypothesis that knowing the sentiment is beneficial in measuring the semantic similarity between words. This metric is used for replacing a feature in test set with a similar feature in the training set that, in turn, helps to improve the performance of supervised sentiment classification of reviews.

Chapter 10 gives the conclusions and a pointer to future works, followed by submissions and references.

Chapter 2

2. LITERATURE SURVEY

This section presents a generic literature survey, covering all the classical works in this domain. The works have been divided into two broad categories: a) Machine Learning Approaches and b) Cognitive Approaches mainly through the works of Janyce Wiebe. We also present a list of features most commonly used in SA.

2.1 FEATURES FOR SENTIMENT ANALYSIS

Feature engineering is an extremely basic and essential task for Sentiment Analysis. Converting a piece of text to a feature vector is the basic step in any data driven approach to SA. In the following section we will see some commonly used features used in Sentiment Analysis and their critiques.

Term Presence vs. Term Frequency

Term frequency has always been considered essential in traditional Information Retrieval and Text Classification tasks. But Pang-Lee *et al.* (2002) found that *term presence* is more important to Sentiment analysis than *term frequency.* That is, binary-valued feature vectors in which the entries merely indicate whether a term occurs (value 1) or not (value 0). This is not counter-intuitive as in the numerous examples we saw before that the presence of even a single string sentiment bearing words can reverse the polarity of the entire sentence. It has also been seen that the occurrence of rare words contain more information than frequently occurring words, a phenomenon called *Hapax Legomena*.

Term Position

Words appearing in certain positions in the text carry more sentiment or weightage than words appearing elsewhere. This is similar to IR where words appearing in topic Titles, Subtitles or Abstracts *etc* are given more weightage than those appearing in the body. In the example given in Section 1.3.c, although the text contains positive words throughout, the presence of a negative sentiment at the end sentence plays the deciding role in determining the sentiment. Thus generally words appearing in the 1st few sentences and last few sentences in a text are given more weightage than those appearing elsewhere.

N-gram Features

N-grams are capable of capturing context to some extent and are widely used in Natural Language Processing tasks. Whether higher order n-grams are useful is a matter of debate. Pang *et al.* (2002) reported that unigrams outperform bigrams when classifying movie reviews by sentiment polarity, but Dave *et al.* (2003) found that in some settings, bigrams and trigrams perform better.

Subsequence Kernels

Most of the works on Sentiment Analysis use word or sentence level model, the results of which are averaged across all words/sentences/n-grams in order to produce a single model output for each review. Bikel *et al.* (2007) use *subsequences.* The intuition is that the feature space implicitly captured by subsequence kernels is sufficiently rich to obviate the need for explicit knowledge engineering or modeling of word- or sentence-level sentiment.

Word sequence kernels of order n are a weighted sum over all possible word sequences of length n that occur in both of the strings being compared.

Mathematically, the word sequence kernel is defined as

$$K_n(s,t) = \sum_{u \in \Sigma^\star} \sum_{\mathbf{i}:s[\mathbf{i}]=u} \sum_{\mathbf{j}:t[\mathbf{j}]=u} \lambda^{(i[n]-i[1]+1)+(j[n]-j[1]+1)}$$

Equation 2.1: Sequence Kernel

where λ is a kernel parameter that can be thought of as a gap penalty, **i** refers to a vector of length n that consists of the indices of string s that correspond to the subsequence u. And, the value $i[n] - i[1] + 1$ can be regarded as the total length of the span of s that constitutes a particular occurrence of the subsequence u. Following Rousu *et al.* (2005), they combine the kernels of orders one through four through an exponential weighting,

$$K(s,t) = \sum_{i=1}^{N} \mu^{1-i} K_i(s,t)$$

Equation 2.2: Combining Sequential Kernels of Different Order

Parts of Speech

Parts of Speech information is most commonly exploited in all NLP tasks. One of the most important reasons is that they provide a crude form of word sense disambiguation.

Adjectives only

Adjectives have been used most frequently as features amongst all parts of speech. A strong correlation between adjectives and subjectivity has been found. Although all the parts of speech are important people most commonly used adjectives to depict most of the sentiments and a high accuracy have been reported by all the works concentrating on only adjectives for feature generation. Pang Lee *et al.* (2002) achieved as accuracy of around 82.8% in movie review domains using only adjectives in movie review domains.

	Proposed Word Lists
Human 1	positive: dazzling, brilliant, phenomenal, excellent, fantastic negative: suck, terrible, awful, unwatchable, hideous
Human 2	positive: gripping, mesmerizing, riveting, spectacular, cool, awesome, thrilling, badass, excellent, moving, exciting negative: bad, clichéd, sucks, boring, stupid, slow

Table 2.1: Word List containing Positive and Negative Adjectives

Adjective-Adverb Combination

Most of the adverbs have no prior polarity. But when they occur with sentiment bearing adjectives, they can play a major role in determining the sentiment of a sentence. Benamara *et al.* (2007) have shown how the adverbs alter the sentiment value of the adjective that they are used with. Adverbs of degree, on the basis of the extent to which they modify this sentiment value, are classified as:

- Adverbs of affirmation: certainly, totally
- Adverbs of doubt: maybe, probably
- Strongly intensifying adverbs: exceedingly, immensely
- Weakly intensifying adverbs: barely, slightly
- Negation and minimizers: never

The work defined two types of AACs:

1. Unary AACs: Containing one adverb and one adjective. The sentiment score of the adjective is modified using the adverb adjoining it.
2. Binary AACs: Containing more than one adverb and an adjective. The sentiment score of the AAC is calculated by iteratively modifying the score of the adjective as each

adverb gets added to it. This is equivalent to defining a binary AAC in terms of two unary AACs iteratively defined.

To calculate the sentiment value of an AAC, a score is associated with it based on the score of the adjective and the adverb. Certain axiomatic rules are specified to specify the way the adverbs modify the sentiment of the adjective. One such axiom can be stated as:

'Each weakly intensifying adverb and each adverb of doubt has a score less than or equal to each strongly intensifying adverb / adverb of affirmation.'

Then, certain functions are described in order to quantify the axioms. A function f takes an adjective-adverb pair and returns its resultant score.

1. *Affirmative and strongly intensifying adverbs*

- AAC-1. If $sc(adj) > 0$ and $adv \in AFF \cup STRONG$, then $f(adv, adj) \geq sc(adj)$.
- AAC-2. If $sc(adj) < 0$ and $adv \in AFF \cup STRONG$, then $f(adv, adj) \leq sc(adj)$.

For example, f value for 'immensely good' is more positive than the score for the positive adjective 'good'.

2. *Weakly intensifying adverbs*

- AAC-3. If $sc(adj) > 0$ and $adv \in WEAK$, then $f(adv, adj) \leq sc(adj)$.
- AAC-4. If $sc(adj) < 0$ and $adv \in WEAK$, then $f(adv, adj) \geq sc(adj)$.

For example, f value for 'barely good' is more negative than the score for the positive adjective 'good'. This is the effect that the weakly intensifying adverb has.

3. *Adverbs of doubt*

- AAC-5. If $sc(adj) > 0$, $adv \in DOUBT$, and $adv' \in AFF \cup STRONG$, then $f(adv, adj) \leq f(adv', adj)$.
- AAC-6. If $sc(adj) < 0$ is negative, $adv \in DOUBT$, and $adv' \in AFF \cup STRONG$, then $f(adv, adj) \geq f(adv', adj)$.

For example, f value for 'probably good' is less than 'immensely good'

4. *Minimizers*

AAC-7. If $sc(adj) > 0$ and $adv \in MIN$, then $f(adv, adj) \leq sc(adj)$.

- AAC-8. If $sc(adj) < 0$ and $adv \in MIN$, then $f(adv, adj) \geq sc(adj)$.

For example, 'hardly good' is less positive than the positive adjective 'good'

Scoring algorithms

a. *Variable scoring algorithm:*

The algorithm modifies the score of the AAC using the function f defined as follows:

- If $adv \in AFF \cup STRONG$, then:

  $$f_{\mathsf{VS}}(adv, adj) = sc(adj) + (1 - sc(adj)) \times sc(adv)$$

  if $sc(adj) > 0$. If $sc(adj) < 0$,

  $$f_{\mathsf{VS}}(adv, adj) = sc(adj) - (1 - sc(adj)) \times sc(adv).$$

- If $adv \in WEAK \cup DOUBT$, VS reverses the above and returns

  $$f_{\mathsf{VS}}(adv, adj) = sc(adj) - (1 - sc(adj)) \times sc(adv)$$

  if $sc(adj) > 0$. If $sc(adj) < 0$, it returns

  $$f_{\mathsf{VS}}(adv, adj) = sc(adj) + (1 - sc(adj)) \times sc(adv).$$

Algorithm 2.1: Scoring Algorithm for Adjective Adverb Combination

Thus, the resultant score of the AAC is the score of the adjective which is suitably adjusted with the effect of the adverb.

b. *Adjective priority scoring algorithm:*

- If $adv \in AFF \cup STRONG$, then

  $$f_{\mathsf{APS}^r}(adv, adj) = \mathsf{min}(1, sc(adj) + r \times sc(adv)).$$

  if $sc(adj) > 0$. If $sc(adj) > 0$,

  $$f_{\mathsf{APS}^r}(adv, adj) = \mathsf{min}(1, sc(adj) - r \times sc(adv)).$$

- If $adv \in WEAK \cup DOUBT$, then APS^r reverses the above and sets $f_{\mathsf{APS}^r}(adv, adj) = \mathsf{max}(0, sc(adj) - r \times sc(adv))$. if $sc(adj) > 0$. If $sc(adj) < 0$, then $f_{\mathsf{APS}^r}(adv, adj) = \mathsf{max}(0, sc(adj) + r \times sc(adv))$.

They give priority to adjectives over the adverbs and modify the score of the adjective by a weight r. This weight r decides the extent to which an adverb influences the score of an adjective.

c. *Adverb priority scoring algorithm*

- If $adv \in AFF \cup STRONG$, then

$$f_{\mathsf{AdvFS}^r}(adv, adj) = \min(1, sc(adv) + r \times sc(adj))$$

if $sc(adj) > 0$. If $sc(adj) < 0$,

$$f_{\mathsf{AdvFS}^r}(adv, adj) = \max(0, sc(adv) - r \times sc(adj)).$$

Topic-Oriented Features

Bag-of-words and phrases are extensively used as features. But in many domains, individual phrase values bear little relation with overall text sentiment. A challenge in Sentiment Analysis of a text is to exploit those aspects of the text which are in some way representative of the tone of the whole text. Often misleading phrases (thwarted expectations) are used to reinforce sentiments. Using bag-of-words or individual phrases will not be able to distinguish between what is said locally in phrases and what is meant globally in the text like drawing of contrasts between the reviewed entity and other entities, sarcasm, understatement, and digressions, all of which are used in abundance in many discourse domains. These features were developed by Turney *et al.* (2002).

1 Semantic Orientation with PMI

Semantic Orientation (SO) refers to a real number measure of the positive or negative sentiment expressed by a word or phrase. The *value phrases* are phrases that are the source of SO values. Once the desired value phrases have been extracted from the text, each one is assigned an SO value. The SO of a phrase is determined based upon the phrase's *pointwise mutual information* (PMI) with the words"excellent" and"poor". PMI is defined by Church and Hanks (1989) as follows:

$PMI(w_1,w_2)=log_2(p(w_1 \ \& \ w_2)/p(w_1)p(w_2))$.

The SO for a phrase is the difference between its PMI with the word"excellent" and its PMI with the word"poor." i.e

SO (phrase)=PMI (phrase,"***excellent***")- PMI (phrase,"***poor***")

Intuitively, this yields values above zero for phrases with greater PMI with the word"excellent" and below zero for greater PMI with"poor". A SO
value of zero would indicate a completely neutral semantic orientation.

	First Word	Second Word	Third Word (Not Extracted)
1.	JJ	NN or NNS	anything
2.	RB, RBR or RBS	JJ	not NN nor NNS
3.	JJ	JJ	not NN nor NNS
4.	NN or NNS	JJ	not NN or NNS
5.	RB, RBR or RBS	VB, VBD, VBN or VBC	anything

Table 2.2: Phrase Patterns Used for Extracting Value Phrases - Turney (2002)

Osgood semantic differentiation with WordNet

WordNet relationships are used to derive three values pertinent to the emotive meaning of adjectives. The three values correspond to the *potency* (strong or weak), *activity* (active or passive) and the *evaluative* (good or bad) factors introduced in Charles Osgood's Theory of Semantic Differentiation (Osgood *et al.*, 1957). These values are derived by measuring the relative minimal path length (MPL) in WordNet between the adjective in question and the pair of words appropriate for the given factor. In the case of the *evaluative* factor (EVA) for example, the comparison is between the MPL between the adjective and"good" and the MPL between the adjective and"bad". The values can be averaged over all the adjectives in a text, yielding three real valued feature values for the text.

"Sentiment expressed with regard to a particular subject can best be identified with reference to the subject itself", Natsukawa and Yi (2003)

In some application domains, it is known in advance what the topic is toward which sentiment is to be evaluated. This can be *exploited* by creating several classes of features based upon the SO values of phrases given their position in relation to the topic of the text. In opinionated texts there is generally a single primary subject about which the opinion is favorable or unfavorable. But secondary subjects are also useful to some extent.

Ex: Opinion (reference) to author in a book review may be useful in a book review.
Ex: In a product review, the attitude towards the company which manufactures the product may be pertinent.

The work considers the following classes of features:

a. *Turney Value*

The average value of all value phrases' SO values for the text

b. *In sentence with THIS WORK*

The average value of all value phrases which occur in the same sentence as a reference to the work being reviewed

c. *Following THIS WORK*

The average value of all value phrases which follow a reference to the work being reviewed directly, or separated only by the copula or a preposition

d. *Preceding THIS WORK*

The average value of all value phrases which precede a reference to the work being reviewed directly, or separated only by the copula or a preposition

e. *In sentence with THIS ARTIST*

With reference to the artist

f. *Following THIS ARTIST*

With reference to the artist

g. *Preceding THIS ARTIST*

With reference to the artist

h. *Text-wide EVA*

The average EVA value of all adjectives in a text

i. *Text-wide POT*

The average POT value of all adjectives in a text

j. *Text-wide ACT*

The average ACT value of all adjectives in a text

k. *TOPIC-sentence EVA*

The average EVA value of all adjectives which share a sentence with the topic of the text

l. *TOPIC-sentence POT*

The average POT value of all adjectives which share a sentence with the topic of the text

m. *TOPIC-sentence ACT*

The average ACT value of all adjectives which share a sentence with the topic of the text

2.2 MACHINE LEARNING APPROACHES

In his work, Pang Lee *et al.* (2002, 2004), compared the performance of Naïve Bayes, Maximum Entropy and Support Vector Machines in SA on different features like considering only unigrams, bigrams, combination of both, incorporating parts of speech and position information, taking only adjectives *etc*. The result has been summarized in the table below.

Features	N umber of Features	Frequency or Presence?	NB	ME	SVM
Unigrams	16165	Freq.	**78.7**	N/A	72.8
Unigrams	16165	Pres.	81.0	80.4	**82.9**
Unigrams+bigrams	32330	Pres.	80.6	80.8	**82.7**
Bigrams	16165	Pres.	77.3	**77.4**	77.1
Unigrams+POS	16695	Pres.	81.5	80.4	**81.9**
Adjectives	2633	Pres.	77.0	**77.7**	75.1
Top 2633 unigrams	2633	Pres.	80.3	81.0	**81.4**
Unigrams+position	22430	Pres.	81.0	80.1	**81.6**

Table 2.3: Accuracy Comparison of Different Classifiers in SA on Movie Review Dataset

It is observed from the results that:

a. Feature presence is more important than feature frequency.
b. Using Bigrams the accuracy actually falls.
c. Accuracy improves if all the frequently occurring words from all parts of speech are taken, not only Adjectives.
d. Incorporating position information increases accuracy.
e. When the feature space is small, Naïve Bayes performs better than SVM. But SVM's perform better when feature space is increased.
f. When feature space is increased, Maximum Entropy may perform better than Naïve Bayes but it may also suffer from overfitting.

Bikel *et al.* (2007) implemented a Subsequence Kernel based Voted Perceptron and compares its performance with standard Support Vector Machines. It is observed that as the number of true positives increase, the increase in the number of false positives is much less in Subsequence Kernel based voted Perceptrons compared to the bag-of-words based SVM's where the increase in false positives with true positives is almost linear. Their model, despite being trained only on the extreme one and five star reviews, formed an excellent continuum over reviews with intermediate star ratings, as shown in the figure below. The authors

comment that “It is rare that we see such behavior associated with lexical features which are typically regarded as discrete and combinatorial. Finally, we note that the voted perceptron is making distinctions that humans found difficult…”.

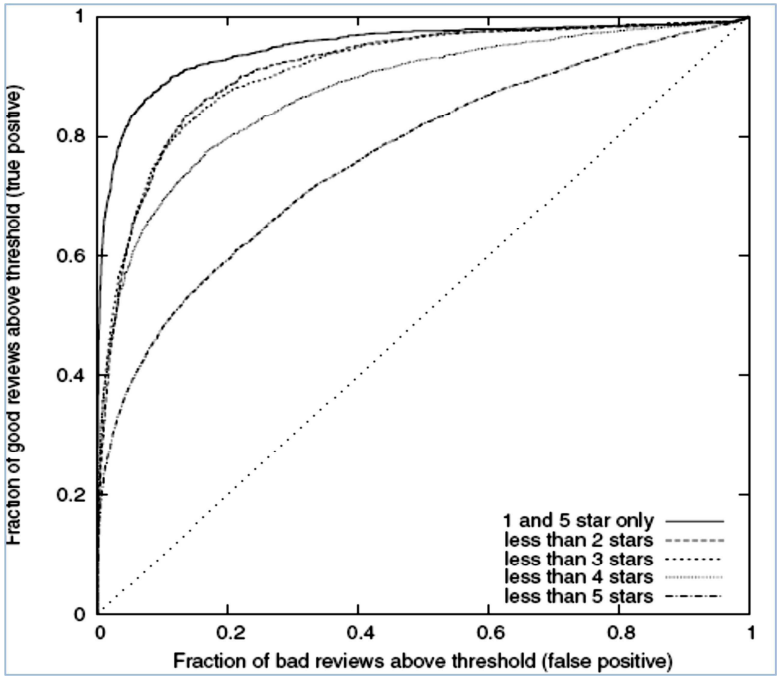


Figure 2.1: Ratio of True Positives and False Positives using Subsequence Kernel based Voted Perceptrons

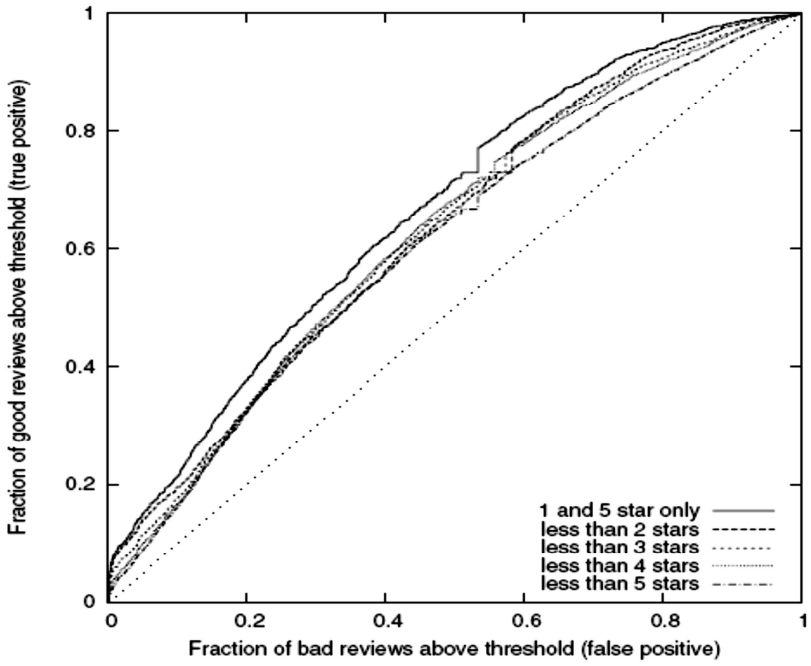


Figure 2.2: Ratio of True Positives and False Positives using Bag-of-Features SVM

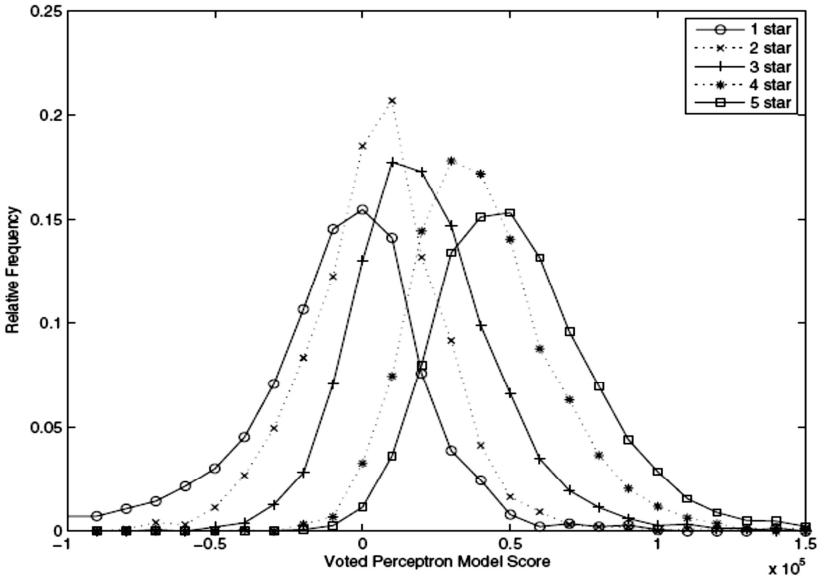


Figure 2.3: Distribution of Voted Perceptron Model Scores by Number of Stars

Pande, Iyer *et al.* performs a detailed comparison of the different classifiers in two phases under two settings. In phase 1, the classifiers are made to distinguish between subjective and objective documents. In phase 2, the classifiers are made to classify positive from negative documents filtered by phase 1. In each phase classifiers have been tested without and with boosting to enhance performance. The classifiers tested are Bayesian Logistic Regression (BLR) with Gaussian and Laplacian prior, Naïve Bayes, Support Vector Machines with linear, polynomial and radial basis functions kernels and Voted Perceptrons.

Classifier	Avg F1	Avg Acc.	Max Acc.
Naïve Bayes	0.4985	78.66	81.55
SVM(Linear)	0.4700	80.83	83.42
SVM(Poly)	0.4536	79.71	81.97
SVM(RBF)	0.0864	78.05	79.4
Voted Perceptron	0.4300	78.64	83.66

Table 2.4: Accuracy Comparison of Different Classifiers, without Boosting (Skewed Dataset)

Classifier	Avg F1	Avg Acc.	Max Acc.
BLR(Gauss)	0.515	80.09	85.51
BLR(Laplace)	0.510	79.98	87.04
Naïve Bayes	0.563	75.63	84.18
SVM(Linear)	0.557	80.2	85.44
SVM(Poly)	0.499	79.22	83.37
Voted Perceptron	0.508	78.70	83.62

Table 2.5: Accuracy Comparison of Different Classifiers, with Boosting (Skewed Dataset)

Classifier	Avg F1	Avg Acc.	Max Acc.
Naïve Bayes	0.882	83.91	84.80
SVM(Linear)	0.880	83.29	85.736
SVM(Poly)	0.8571	80.80	82.946
SVM(RBF)	0.757	68.22	74.573
Voted Perceptron	0.875	82.96	85.43

Table 2.6: Accuracy Comparison of Different Classifiers, without Boosting (Balanced Dataset)

In phase 1, it is observed that:

1. Accuracy very high but F1 scores are low due to skewed data (number of objective samples were 10,000 whereas number of polar samples were 25000) due to which either recall is low or false-positive rate is considerably high.
2. Without boosting, undersampling or oversampling maximum recall attained was 65%.
3. Lowest recall was attained for feature based on Gain Ratio and/or SVM with RBF kernel.
4. With boosting highest recall attained was 85% with false negative rate 30%.
5. Naïve Bayes and SMO gave the best recall.
6. Voted Perceptrons, BLR gave less than 65% recall
7. Voted Perceptrons and SVM(linear) give the best accuracy.
8. It is astonishing to see SVM(RBF) give the lowest accuracy. The reason may be overfitting.

In phase 2, it is observed that:

1. SVM with RBF again performs badly.
2. SVM (linear) and Voted Perceptrons again have the best accuracy. They are found to give good results with both Information gain, Chi-Square feature selection methods.
3. Naïve Bayes favors Chi-Square over Information Gain.
4. Boosting does not improve performance much.

Mullen *et al.* (2004) used Support Vector Machines with diverse information measures by using features like the PMI, Lemma, Turney, Osgood values along with other topic oriented features.

It is observed that:

1. Using only Turney values, a high accuracy can be achieved.
2. The addition of Osgood values does not seem to yield improvement in any of the models.
3. Using only Lemmas instead of Unigrams result in a much better performance.

4. The inclusion of all PMI values with lemmas outperforms the use of only the Turney values, suggesting that the incorporation of the available topic relations is helpful.
5. The best performance is achieved by using Lemmas and PMI or Osgood Values.

Model	5 folds	10 folds	20 folds	100 folds
Turney Values only	72%	73%	72%	72%
All (THIS WORK and THIS ARTIST) PMI	70%	70%	68%	69%
THIS WORK PMI	72%	69%	70%	71%
All Osgood	64%	64%	65%	64%
All PMI and Osgood	74%	71%	74%	72%
Unigrams	79%	80%	78%	82%
Unigrams, PMI, Osgood	81%	80%	82%	82%
Lemmas	83%	85%	84%	84%
Lemmas and Osgood	83%	84%	84%	84%
Lemmas and Turney	84%	85%	84%	84%
Lemmas, Turney, text wide Osgood	84%	85%	84%	84%
Lemmas, PMI, Osgood	84%	85%	84%	86%
Lemmas and PMI	84%	85%	85%	86%
Hybrid SVM (PMI/Osgood and Lemmas)	86%	87%	84%	89%

Table 2.7: Accuracy Comparison of Different Features on SVM using a Linear Kernel

2.3 COGNITIVE APPROACHES (DISCOURSE)

2.3.1 What is Subjectivity Analysis, Perspective and Narratives?

The input to the Sentiment Classifier is always an opinionated text *i.e.* a text containing positive or negative sentiment. Thus one needs to filter out objective facts from a text and subject only the opinions to the Sentiment Classifier. This work of extracting or filtering out the objective facts from subjective opinions is called Subjectivity Analysis.

A piece of text often contains other person's point of view or accounts from a third person which contain a gamut of emotions or opinions from different characters or perspectives. Thus it is important to identify the character to who an opinion can be attributed to. Thus the objective here is not only to detect whether a piece of text is opinionated or not but also who is responsible for that opinion.

A *narrative* is a story that is created in a constructive format. It describes a sequence of fictional and non-fictional events. A narrative can also be told by a character within a larger narrative. It is the fiction-writing mode whereby the narrator communicates directly to the reader.

A *perspective* is the point of view. Narrative is told from the perspective of one or more of its characters. It can also contain passages not told from the perspective of any character.

Subjective sentences *portray* a character's *thoughts* – represented thoughts or present a *scene perceived* by the character – represented perception, private state such as seeing, wanting or feeling ill – that is some perceptual, psychological or experiential state not open to objective observation or verification.

Objective sentences present the story *directly,* rather than through *thoughts or perceptions* of a character.

Thus subjective sentences take a character's psychological point of view or POV (Uspensky 1973). For narrative understanding it is absolutely essential to track the POV as it distinguishes between beliefs of characters and facts of the story.

2.3.2 Discourse-Level Analysis

But in order to identify the character and its perspective a sentence level analysis will not do. This demands a discourse level analysis as the sentences are not always explicitly marked by subjective elements and the subjective sentences do not directly imply the subjective character, Wiebe *et al.* (1991).

Example: [1.1]He wanted to talk to Dennys. [1.2]How were they going to be able to get home from this strange desert land into which they had been cast and which was heaven knew where in all the countless solar systems in all the countless galaxies? [L'Engle, Many Waters, p. 91]

Sentence (1.2) is a represented thought, and (2.2) is a represented perception, presenting what the character sees as he sees it, yet neither is explicitly marked as such. Also, neither indicates who the subjective character is.

Subjective Sentences which do no contain any subjective elements or subjective character appear in the midst of other subjective sentences attributed to the same subjective character. Thus subjectivity needs to be determined at discourse level. Instead of subjective or objective sentences we have subjective and objective contexts consisting of 1 or more subjective or objective sentences attributed to the same subjective character or objective sentences.

There are regularities in a way the author initiate, continue or resumes a character's POV. Certain combination of the sentence features (like tense, aspect, lexical items expressing subjectivity, identity of actors or experiencers of those states of affairs) and the current context (like whether the previous sentence was subjective or objective or whether there was a scene break or paragraph break) helps in detecting the continuity of the POV of a character.

The POV tracking needs an extensive discourse analysis. For example, the use of a full noun when a pronoun would have sufficed denotes that change in POV has occurred. On the other hand use of anaphoric pronoun denotes continuity in current POV.

The private state reports of a character, that expresses whether the subjective character is ill or angry, can only be reported. Now, if"*John was furious"* is the subjective statement of a character Mary, it is only her *represented thought or opinion* about John. Thus the distinction between private state report and represented thought is essential for discourse processing. This is because, as the subjective character is always the subject of a private state report, pronouns can be used to refer him despite references to some other entity of same number and gender. But in a represented thought the referent of a pronoun can be someone in an earlier represented thought.

Example:

"*Dwayne wasn't sure what John was scared of. What in the arcade could scare a boy like that? He could see tear's in John's eyes. He could tell they were tears because .... Maybe that was why he was crying.*

"I want to leave", he said."

Here the last sentence is objective but the previous sentences are subjective sentences of Dwayne. 'He' in the last sentence refers to Dwayne even though John is the last previously mentioned entity. This suggests there is a change of POV and discourse analysis is required to detect it.

2.3.3 Subjective Contexts

Recognition of a subjective context requires the presence of linguistic signals. Wiebe *et al.* (1988) recognizes the following subjective signals..

a. Psychological Verbs, Actions, Adjectives and Perceptual Verbs

1. Psychological verbs (e.g. 'think', 'wonder', 'realize', 'went')
2. Perceptual verbs (e.g.. 'see', 'hear')
3. Psychological adjectives (e.g., 'delighted', 'happy', 'jealous', 'scared')
4. Psychological actions- (e.g.,"he smiled","she gasped","she winced"

a. Subjective Elements

1. Exclamations, which express emotions
2. Questions, which express wonder

3. Epithets, such as 'the bastard', which express some qualification of the referent.
4. Kinship *terms,* e.g., 'Daddy', 'Morn', and 'Aunt Margaret', which express a relationship to the referee.
5. Evaluative adjectives, which express an attitude toward the referent, e.g., 'ghastly', 'surprising', 'poor', and 'damned',
6. Intensifiers such as 'too', 'quite', and 'so' are also evaluative
7. Emphasizers like"really","just"

b. More Subjective Elements

1. Modal verbs of obligation, possibility, and necessity. For example, 'should' is a modal verb of obligation)
2. *content (or attitudinal) disjuncts* which comment on the content of the utterance. For example,"*likely',* 'maybe', 'probably', and 'perhaps' express some degree of doubt
3. Conjuncts, which comment on the connection between items. For example, 'anyhow', 'anyway', 'still', and 'after all' express concession
4. Uses of 'this', 'that', 'these', and 'those' that Robin Lakoff (1974) has identified as *emotional deixis.*

In conversation, they are"generally linked to the speaker's emotional involvement in the subject-matter of his utterance" (Lakoff 1974: 347); in thlrd-person narrative, they are linked to the subjective character's emotional involvement in the subject matter of his thoughts or perceptions.

Examples:

[2.1]She [Hannah] ***winced*** *as she heard them crash to the platform.*

[3.1]He could tell they were tears because his eyes were ***too*** *shiny. Too round.*

[4.1]Jody managed a frail smile. [4.2]She was ***a little*** *bit ashamed. [4.3] She should* ***really*** *try to be more cheerful for Aunt Margaret's sake. [4.4]****After all****, Aunt Margaret had troubles of her own--she was the mother of that ghastly Dill.*

In the above examples, the words in bold are subjective elements. The presence of a subjective element indicates the presence of a subjective character.

2.3.4 Identifying a Subjective Character

Subjective Character can sometimes be directly identified in a sentence (for example when there is a narrative parenthetical), Wiebe *et al.* (1990).

If it is not identifiable, then it is one of the 2 previously mentioned characters-

1. the subjective character of the previous subjective sentence

It either continues the POV of the subjective character or resumes it.

2. actor of an action denoted by a previous objective sentence

Examples:

[5.1]Why, Jake,you lazy bean," Augustus said, [5.2] and walked off. [5.3] Jake had a stubborn streak in him, [5.4]and once it was activated even Call could seldom do much with him.

(5.3) and (5.4) represent the point of view of Augustus, the actor of an action denoted by a previous objective sentence, (5.1). But the last subjective character is Jake, so Augustus's point of view is initiated, not merely resumed or continued.

In order to identify the subjective character one needs to keep track of expected subjective characters encountered. However drastic spatial and temporal discontinuities can block the continuation or resumption of a character's POV.

Ex: *scene break, paragraph break*

The subjective character may also be identified from a private-state sentence. It *can* be the experiencer of a private state sentence. ***Exception*** occurs when a subjective sentence is followed by a private state report without paragraph break where the experiencer is different from the subjective character.

Example of a scene break:

[6.l]Drown me?" Augustus said. [6.2] Why if' anybody had tried it, those girls would have clawed them to shreds." [6.3] He knew Call was mad, [6.4] but wasn't much inclined to humor him. [6.5] It was his dinner table as much as Call's, [6.6] and if Call didn't like the conversation he could go to bed.

[6.7] Call knew there was no point in arguing. [6.8] That was what Augustus wanted: argument. [6.9] He didn't really care what the question was, [6.10] and it made no great difference to him which side he was on. [6.11] He just plain loved to argue.

Sentences (6.1)-(6.2) are Augustus's subjective sentences and (6.7)-(6.11) are Call's. So, (6.7) initiates a new point of view. It is a private-state sentence and the subjective character, Call, is the experiencer of the private state denoted.

Subjective character of a private state report is always the experiencer which can be directly determined from the sentence. But such cannot be said in case of a represented thought. If private-state report is indicated to be a represented thought, then subjective character is the expected subjective character. But an important aspect is the scope of the subjective element. Consider the following examples.

[7.1] Japheth, ***evidently*** *realizing that they were no longer behind him, turned around [7.2]and jogged back toward them, seemingly cool and unwinded.*
[8.1]Urgh! She) the [girl] thought. [8.2]How could the poor thing have married him in the first place?
[8.3]Johnnie Martin could not believe that he was seeing that old bag's black eyes sparkling with disgust and unsheathed contempt at him.

Sentence (8.3) is a private-state report and the subjective character is the experiencer (Johnnie Martin). This is so even though (8.3) contains the subjective element 'old bag' and even though there is an expected subjective character (the girl) when it is encountered. Because 'old bag' appears within the scope of the private-state term 'believe', it is not considered in identifying the subjective character. On the other hand, the subjective clement 'evidently ' in (7.1) is not in the scope of 'realizing' (*i.e.*, it is *non-subordinated),* so it can be used to identify the subjective character.

---

if *the sentence contains a narrative parenthetical then*
    SC is the subject of the parenthetical
else if *the sentence is a private-state sentence then*
    if *it has a non-subordinated subjective clement*
    or the text situation is continuing-subjective then
        SC is identified from the previous context
    else *SC is the experiencer*
    end if
else
    SC is identified from the previous context
end if

Algorithm 2.2: To Identify the Subjective Character

__if__ there are two expected subjective characters then

__if__ the sentence is about the last active character then

SC is the last subjective character

__else__ SC is the last active character

__end if__

__else if__ there is an expected subjective character then

SC is the expected subjective character

__else__ SC is unidentified

__end if__

Algorithm 2.3: To Identify the Subjective Character from Previous Context

2.3.5 Identifying Perspective in Narrative

A belief space is accessed by a stack of individuals. It consists of what the bottom member of the stack believes that what the top member believes.

The reader is always the bottom member. The belief space corresponding to a stack consisting only of the reader contains the set of propositions that the reader believes are true. The CP determines the current belief space with respect to which references are understood.

__if__ 'X' is an indefinite noun phrase of the form 'a Y' then

create a new concept, N; build in CP's belief space the proposition that N is a Y;

return N

__else if__ 'X' is a definite noun phrase or proper name,

if a proposition that N is X can be found in the CP's belief space,

return N

else if a proposition that N is X can be found in a belief space other than the

CP's, then add the found proposition to the CP's belief space;

return N

else create a new concept, N; build in CP's belief space the proposition that N is X;

return N

Algorithm 2.4: Algorithm for Understanding a Non-Anaphoric, Specific Reference 'X' in Third-Person Narrative

If a reference is a subjective element, such as 'the bastard', it cannot be understood entirely propositionally, since it expresses subjectivity. How it should be understood depends on the particular subjective element.

It does not understand anaphoric references. However, anaphor comprehension can be affected by perspective.

Examples:

[9.1]The man had turned. [9.2]He started to walk away ***quickly*** *in the direction of the public library.*

[9.3]"O.K.," said Joe,"get Rosie."

[9.4]Zoe crept back to the blinker. [9.5]She felt hollow in her stomach. [9.6]She'd never ***really*** *expected to see the Enemy again. [Ones], War Work, p. 64]*

'The Enemy' is an anaphoric reference that occurs in a subjective context (established by (9.5), which is a Psychological report). It co-specifies 'the man' in (9.1) and 'He' in (9.2). It reflects Zoe's belief that the man is an enemy spy, although it is not at all clear to the reader, at point' that he is.

Personal pronouns can also reflect the beliefs of a character.

Assertive indefinite pronouns ex. 'someone', 'something', 'somebody' refer to particular people, things, *etc*., without identifying them. When referring to a particular referent, a speaker typically uses an assertive indefinite pronoun if

(1) she doesn't know the identity of the referent

(2) She doesn't want the addressee to know the identity of the referent, or

(3) she doesn't believe that the identity of the referent is relevant to the conversation.

A character's thoughts end perceptions are not directed toward an addressee, and so the first of these uses is the predominant one in subjective contexts.

Example:[10] Suddenly she [Zoe] gasped. She had touched ***somebody****! [O'Neal, War Work, p. 129]*

There is no explicit statement in the novel that Zoe does not know whom she touched; this has to be inferred from the use of 'somebody'.

Definite references are used only if the speaker believes that the addressee has enough information to interpret them. Specific indefinite references are used in a subjective context when the referent is unfamiliar to the subjective character. However, the referent may not be unknown to the reader or to the other characters.

[11] There they [the King and his men] saw close beside them a great rubbleheap; and suddenly they were aware of ***two small figures*** *lying on it at their ease, grey-clad, hardly to be seen among the stones. [Tolkien, The Two Towers, p. 206]*

The reader knows that the King and his men have come upon two hobbits, Merry end Pippin. The King and his men do not know the hobbits, but other characters also present in the scene do know them. When the King and his man are on the top of the CP (after 'saw' and continued by 'were aware of'), the hobbits are not referred to by name, but as 'two small figures'. Thus new referents are created and propositions are built in the belief space of the King and his men that they are small figures. The new referents can be asserted to be co-extensional with the concepts who the reader and other characters believe are named 'Merry' and 'Pippin'.

2.3.6 Evaluation

The algorithms were tested on 450 sentential input items (exclusive of paragraph and scene breaks) from each of two novels, *Lonesome Dove* by Larry McMurtry and *The Magic of the Glits* by Carole S. Adler. *Lonesome Dove* is an adult novel that has many subjective characters, and *The Magic of the Glits* is a children's' novel that has one main subjective character. The input items are those of the complete sentences of every fifth page of these novels, starting in *Lonesome Dove* with page 176 and ending with page 236 (13 pages total), and starting in *The Magic of the Glits* with page 1 and ending with page 86 (18 pages total). (For each book, the first part of an additional page was used to make the number of input items exactly equal to 450.) Page 176 in *Lonesome Dove* is the beginning of a chapter in the middle of the novel. The earlier pages of the novel were considered during the development of the algorithm.

Interpretation	Actual Instances	Primary Errors	Incorrect Interpretations
<subjective, x>	271/450 (60%)	20/271 (7%)	13 objective 7 (subjective, y), $y \neq x$
Objective	179/450 (40%)	7/179 (4%)	7 (subjective, x)
Objective, other than simple quoted speech	54/450 (12%)	7/54 (13%)	7 (subjective, x)

Table 2.8: Results for Lonesome Drove by Interpretation

Point-of-View Operation	Actual Instances	Primary Errors	Incorrect Interpretations
Continuation	215/450 (48%)	11/215 (5%)	1 initiation 10 objective
Resumption	20/450 (4%)	0/20 (0%)	-
Initiation	36/450 (8%)	9/36 (25%)	5 resumptions 1 initiation 3 objective
Objective	179/450 (40%)	7/179 (4%)	4 continuations 3 resumptions
Objective, other than simple quoted speech	54/450 (12%)	7/54 (13%)	4 continuations 3 resumptions

Table 2.9: Results for Lonesome Drove by Point-of-View Operation

In Lonesome Dove, out of the 450 input items, the algorithm committed 27 primary errors (6%) and 28 secondary errors (6%). Many of the input items, 125 of them (28%), are simple items of quoted speech (*i.e.*, they do not have potential subjective elements in the discourse parenthetical, or subordinated clauses outside the quoted string that have private-state terms, private-state-action terms, or potential subjective elements).

In Table 2.8, the first row, is interpreted as: Out of the 271 actual subjective sentences, the algorithm committed 20 primary errors. It interpreted 13 subjective sentences to be objective, and 7 to be the subjective sentence of the wrong subjective character.

In Table 2.9, the first row, is interpreted as: Out of the 215 items that actually continue a character's point of view, the algorithm committed 11 primary errors. It interpreted 1 of them to be an initiation and 10 to be objective. Notice that the last column of the row for initiations includes an initiation. This means that for one actual initiation, the algorithm was correct that a character's point of view was initiated, but incorrect as to the identity of that character.

Interpretation	Actual Instances	Primary Errors	Incorrect Interpretations
<subjective, x>	125/450 (28%)	12/125 (10%)	10 objective 2 (subjective, y), $y \neq x$
Objective	325/450 (72%)	22/325 (7%)	22 (subjective, x)
objective, other than simple quoted speech	97/450 (22%)	22/97 (23%)	22 (subjective, x)

Table 2.10: Results for The Magic of the Glits by Interpretation

Point-of-View Operation	Actual Instances	Primary Errors	Incorrect Interpretations
Continuation	79/450 (18%)	4/79 (5%)	4 objective
Resumption	41/450 (9%)	7/41 (17%)	2 initiations 5 objective
Initiation	5/450 (1%)	1/5 (20%)	1 objective
Objective	325/450 (72%)	22/325 (7%)	4 continuations 9 resumptions 9 initiations
objective, other than simple quoted speech	97/450 (22%)	22/97 (23%)	4 continuations 9 resumptions 9 initiations

Table 2.11: Results for The Magic of the Glits by Point-of-View Operations

In *The Magic of the Glits,* out of the 450 input items, the algorithm committed 34 primary errors (8%) and 21 secondary errors (5%). There are 228 items that are simple quoted speech (51%). Tables 2.10 and 11 present the kinds of results for this novel that were given above in Tables 2.8 and 2.9 for *Lonesome Dove.*

2.4 SENTIMENT ANALYSIS AT IIT BOMBAY

Balamurali *et al.* (2011) presents an innovative idea to introduce sense based sentiment analysis. This implies shifting from lexeme feature space to semantic space *i.e.* from simple words to their synsets. The works in SA, for so long, concentrated on lexeme feature space or identifying relations between words using parsing. The need for integrating sense to SA was the need of the hour due to the following scenarios, as identified by the authors:

a. A word may have some sentiment-bearing and some non-sentiment-bearing senses
b. There may be different senses of a word that bear sentiment of opposite polarity
c. The same sense can be manifested by different words (appearing in the same synset)

Using sense as features helps to exploit the idea of sense/concepts and the hierarchical structure of the WordNet. The following feature representations were used by the authors and their performance were compared to that of lexeme based features:

a. A group of word senses that have been manually annotated (M)
b. A group of word senses that have been annotated by an automatic WSD (I)

c. A group of manually annotated word senses and words (both separately as features) (Sense + Words(M))
d. A group of automatically annotated word senses and words (both separately as features) (Sense + Words(I))

Sense + Words(M) and Sense + Words(I) were used to overcome non-coverage of WordNet for some noun synsets.

The authors used synset-replacement strategies to deal with non-coverage, in case a synset in test document is not found in the training documents. In that case the target unknown synset is replaced with its closest counterpart among the WordNet synsets by using some metric. The metrics used by the authors were LIN, LESK and LCH.

SVM's were used for classification of the feature vectors and IWSD was used for automatic WSD. Extensive experiments were done to compare the performance of the 4 feature representations with lexeme representation. Best performance, in terms of accuracy, was obtained by using sense based SA with manual annotation (with an accuracy of 90.2% and an increase of 5.3% over the baseline accuracy) followed by Sense(M), Sense + Words(I), Sense(I) and lexeme feature representation. LESK was found to perform the best among the 3 metrics used in replacement strategies.

One of the reasons for improvements was attributed to feature abstraction and dimensionality reduction leading to noise reduction. The work achieved its target of bringing a new dimension to SA by introducing sense based SA.

Aditya *et al.* (2010) introduced Sentiment Analysis to Indian languages namely, Hindi. Though, much work has been done in SA in English, little or no work has been done so-far in Hindi. The authors exploited 3 different approaches to tackling SA in Hindi:

1. They developed a sense annotated corpora for Hindi, so that any supervised classifier can be trained on that corpora.
2. They translated the Hindi document to English and used a classifier trained in English documents to classify it.
3. They developed a sentiment lexicon for Hindi called the Hindi SentiWordNet (H-SWN).

The authors first tried to use the in-language classifier and if training data was not available, they settled for a rough Machine Translation of the document to resource-rich English. In case MT could not be done they used the H-SWN and used a majority voting approach to find the polarity of the document.

As expected, the classifier trained in in-language documents gave the best performance. This was followed by the MT system of converting the source document to English. One of the reasons for its degraded performance can be attributed to the absence of Word Sense Disambiguation, which led to a different meaning of the Hindi word in English after translation. The lexical resource based approach gave the worse performance amongst the 3 approaches. This was mainly due to the coverage of the H-SWN which was quite low.

Aditya *et al.* (2010) took Sentiment Analysis to a new terrain by introducing it to micro-blogs namely, Twitter. They developed the C-Feel-It system that extracted posts called Tweets from the Twitter, related to the user query, and evaluated its sentiment based on 4 lexical resources. Twitter is a very noisy medium where the user posts different forms of slangs, abbreviations, smileys *etc*. There is also a high occurrence of spams generated by bots. Due to these reasons, the accuracy of the system deteriorated mainly because the words in the post were not present in the lexical resources. However, the authors used some form of normalization to compensate somewhat for the inherent noise. They also used a slang and emoticon dictionary for polarity evaluation. The authors used the 4 lexical resources *Taboada, Inquirer, SentiWordNet and Subjectivity Lexicon* and settled for a majority voting for the final polarity of the tweet. This work is novel mainly because it exploits a new domain.

Chapter 3

3. LITERATURE SURVEY FOR CURRENT WORK

This section presents an extensive literature survey on the specific aspects of sentiment analysis covered in our work, namely – *discourse coherency, feature specificity, Twitter-related works in sentiment analysis* and *subjectivity analysis*. Apart from these, we also present a literature survey on some other areas of *information extraction* that are related to our work, namely – *similarty metrics, extractive text summarization, concept expansion using Wikipedia* and *video categorization.*

3.1 DISCOURSE SPECIFIC SENTIMENT ANALYSIS

Marcu (2000) discussed probabilistic models for identifying elementary discourse units at clausal level and generating trees at the sentence level, using lexical and syntactic information from discourse-annotated corpus of RST. Wellner *et al.* (2006) considered the problem of automatically identifying arguments of discourse connectives in the PDTB. They modeled the problem as a predicate-argument identification where the predicates were discourse connectives and arguments served as anchors for discourse segments. Wolf *et al.* (2005) present a set of discourse structure relations and ways to code or represent them. The relations were based on Hobbs (1985). They report a method for annotating discourse coherent structures and found different kinds of crossed dependencies.

In the work, *Contextual Valence Shifters* (Polanyi *et al.,* 2004), the authors investigate the effect of *intensifiers, negatives, modals* and *connectors* that changes the prior polarity or valence of the words and brings out a new meaning or perspective. They also talk about pre-suppositional items and irony and present a simple weighting scheme to deal with them.

Somasundaran *et al*. (2009) and Asher *et al.* (2008) discuss some discourse-based supervised and unsupervised approaches to opinion analysis. Zhou *et al.* (2011) present an approach to identify discourse relations as identified by RST. Instead of depending on cue-phrase based methods to identify discourse relations, they leverage it to adopt an unsupervised approach that would generate semantic sequential representations (SSRs) without cue phrases.

Taboada *et al*. (2008) leverage discourse to identify relevant sentences in the text for sentiment analysis. However, they narrow their focus to adjectives alone in the relevant portions of the text while ignoring the remaining parts of speech of the text.

Most of these discourse based works make use of a *discourse parser* or a *dependency parser* to identify the scope of the discourse relations and the opinion frames. As said before, the parsers fare poorly in the presence of noisy text like *ungrammatical sentences* and *spelling mistakes* (Dey *et al.*, 2009). In addition, the use of parsing slows down any real-time interactive system due to increased processing time. For this reason, the micro-blog applications mostly build on a bag-of-words model.

3.2 FEATURE SPECIFIC SENTIMENT ANALYSIS

Chen *et al.* (2010) uses dependency parsing and shallow semantic analysis for Chinese opinion related expression extraction. They categorize relations as, Topic and sentiment located in the same sub-sentence and quite close to each other (like rule "an adjective plus a noun" is mostly a potential opinion-element relation), Topic and sentiment located in adjacent sub-sentences and the two sub-sentences are parallel in structure (that is to say, the two adjacent sub-sentences are connected by some coherent word, like although/but, and *etc*), Topic and sentiment located in different sub-sentences, either being adjacent or not, but the different sub sentences are independent of each other, no parallel structures any more.

Wu *et al.* (2009) use phrase dependency parsing for opinion mining. In dependency grammar, structure is determined by the relation between a head and its dependents. The dependent is a modifier or complement and the head plays a more important role in determining the behaviors of the pair. The authors want to compromise between the information loss of the word level dependency in dependency parsing as it does not explicitly provide local structures and syntactic categories of phrases and the information gain in extracting long distance relations. Hence they extend the dependency tree node with phrases."

Hu *et al.* (2004) used frequent item sets to extract the most relevant features from a domain and pruned it to obtain a subset of features. They extract the nearby adjectives to a feature as an *opinion word* regarding that feature. Using a seed set of labeled Adjectives, which they manually develop for each domain, they further expand it using WordNet and use them to classify the extracted opinion words as positive or negative.

Lakkaraju *et al.* (2011) propose a joint sentiment topic model to probabilistically model the set of features and sentiment topics using HMM-LDA. It is an unsupervised system

which models the distribution of features and opinions in a review and is thus a generative model.

Most of the works mentioned above require labeled datasets for training their models for each of the domains. If there is a new domain about which no prior information is available or if there are mixed reviews from multiple domains inter-mixed (as in *Twitter*), where the domain for any specific product cannot be identified, then it would be difficult to train the models. The works do not exploit the fact that majority of the reviews have a lot of domain independent components. If those domain independent parameters are used to capture the associations between features and their associated opinion expressions, the models would capture majority of the feature specific sentiments with minimal data requirement.

3.3 SEMANTIC SIMILARITY METRICS

Various approaches for evaluating the similarity between two words can be broadly classified into two categories: edge-based methods and informationcontent-based methods. One of the earliest works in edge-based calculation of similarity is by Rada et al. (1989), where in, they propose a metric"Distance" over a semantic net of hierarchical relations as the shortest path length between the two nodes. This has been the basis for all the metrics involving simple edge-counting to calculate the distance between two nodes. However, the simple edge-counting fails to consider the variable density of nodes across the taxonomy. It also fails to include relationships other than the is-a relationship, thus, missing out on important information in a generic semantic ontology, like WordNet.

In contrast to edge-based methods, Richardson et al. (1994) and Resnik (1995*a*) propose a node-based approach to find the semantic similarity. They approximate conceptual similarity between two WordNet concepts as the maximum information content among classes that subsume both the concepts. Resnik (1995*b*) advanced this idea by defining the information content of a concept based on the probability of encountering an instance of that concept. Alternatively, Wu & Palmer (1994) compare two concepts based on the length of the path between the root of the hierarchy and the least common subsumer of the concepts.

Jiang & Conrath (1997) and Leacock et al. (1998) combine the above two approaches by using the information content as weights for the edges between concepts. They further reinforce the definition of information content of a concept by adding corpus statistical information.

Instead of measuring the similarity of concepts, some other approaches measure their

relatedness. Hirst & St-Onge (1997) introduce an additional notion of direction along with the length of paths for measuring the relatedness of two concepts. Banerjee & Pedersen (2003) and Patwardhan (2003) leverage the gloss information present in WordNet in order to calculate the relatedness of two concepts. Banerjee & Pedersen (2003) assigns relatedness scores based on the overlap between the gloss of the two concepts. Patwardhan (2003) use a vector representation of the gloss, based on the context vector of the terms in the gloss. The relatedness is then the cosine between the gloss vectors of the two concepts.

Our work is most related to the work of Wan & Angryk (2007) which improves on Banerjee & Pedersen (2003) and Patwardhan (2003) by including relations other than the is-a relationship. They use an extended gloss definition for a concept which is defined as the original gloss appended by the gloss of all the concepts related to the given concept. They create concept vectors for each sense based on which they create context vectors which are an order higher to the concept vectors. Finally, they use cosine of the angle between the vectors of the different concepts to find their relatedness. This approach is better than other approaches as it captures the context of the concepts to a much larger extent. However, all these methods lack on a common ground. They fail to incorporate sentiment information in calculating the similarity/relatedness of two concepts. We postulate that sentiment information is crucial in finding the similarity between two concepts.

3.4 SENTIMENT ANALYSIS IN TWITTER

Twitter is a micro-blogging website and ranks second amongst the present social media websites (Prelovac 2010). A micro-blog allows users to exchange small elements of content such as short sentences, individual pages, or video links. Alec *et al.* (2009) provide one of the first studies on sentiment analysis on micro-blogging websites. Barbosa *et al.* (2010) and Bermingham *et al.* (2010) both cite noisy data as one of the biggest hurdles in analyzing text in such media. Alec *et al.* (2009) describes a distant supervision-based approach for sentiment classification. They use hashtags in tweets to create training data and implement a multi-class classifier with topic-dependent clusters. Barbosa *et al.* (2010) propose an approach to sentiment analysis in Twitter using POS-tagged n-gram features and some Twitter specific features like hashtags. Our system is inspired from *C-Feel-IT*, a Twitter based sentiment analysis system (Joshi *et al.*, 2011). However, our system is an enhanced version of their rule based system with specialized modules to tackle Twitter spam, text normalization and entity specific sentiment analysis.

To the best of our knowledge, there has not been any specific work regarding spam filtering for tweets in the context of sentiment analysis. General spam filtering techniques include approaches that implement Bayesian filter (Sahami, 1998; Graham, 2006), or SVM-based filters along with various boosting algorithms to further enhance the accuracies (Drucker *et al.,* 1999).

Twitter is a very noisy medium. However, not much work has been done in the area of text normalization in the social media especially pertaining to Twitter. But there has been some work in the related area of SMS-es. Aw *et al.,* (2006) and Raghunathan *et al.,* (2009) used a MT-based system for text normalization. Choudhury *et al.,* (2007) deployed a HMM for word-level decoding in SMS-es; while Catherine *et al.,* (2008) implemented a combination of both by using two normalization systems: first a SMT model, and then a second model for speech recognition system. Another approach to text normalization has been to consider each word as a corrupt word after being passed through a noisy channel, which essentially boils down to spell-checking itself. Mayes (1991) provide one such approach. Church *et al.,* (1991) provide a more sophisticated approach by associating weights to the probable edits required to correct the word.

We follow the approach of Church *et al.,* (1991) and attempt to infuse linguistic rules within the minimum edit distance (Levenstein, 1966). We adopt this simpler approach due to the lack of publicly available parallel corpora for text normalization in Twitter.

Unlike in Twitter, there has been quite a few works on general entity specific sentiment analysis. Nasukawa *et al.,* (2003) developed a lexicon and sentiment transfer rules to extract sentiment phrases. Mullen *et al.,* (2004) used Osgood and Turney values to extract value phrases, *i.e.* sentiment bearing phrases from the sentence. Many approaches have also tried to leverage dependency parsing in entity-specific SA. Mosha (2010) uses dependency parsing and shallow semantic analysis for Chinese opinion related expression extraction. Wu *et al.,* (2009) used phrase dependency parsing for opinion mining. Mukherjee *et al.* (2012) exploit dependency parsing for graph based clustering of opinion expressions about various features to extract the opinion expression about a target feature. We follow a dependency parsing based approach for entity specific SA as it captures long distance relations, syntactic discontinuity and variable word order, as is prevalent in Twitter.

The works (Alec *et al.*, 2009; Read *et al.*, 2005; Pak *et al.*, 2010; Gonzalez *et al.* (2011)) evaluate their system on a dataset crawled and auto-annotated based on *emoticons*,

hashtags. We show, in this work, that a good performance on such a dataset does not ensure a similar performance in a general setting.

3.5 EXTRACTIVE SUMMARIZATION

There are 2 prominent paradigms in automatic text summarization (Das *et al.*, 2007): *extractive* and *abstractive* text summarization. While *extractive* text summarization attempts to identify prominent sections of a text by giving more emphasis on the content of the summary, *abstractive* text summarization gives more emphasis on the form so that sentences are syntactically and semantically coherent. The *topic-driven* summarization paradigm is more common to IR where the summary content is based on the user query about a particular topic. Luhn (1958) attempts to find the top-ranked significant sentences based on the frequency of the content words present in it. Edmundson (1969) gives importance to the position of a sentence *i.e.* where the sentence appears in the text and comes up with an optimum position policy and emphasis on the cue words. Aone *et al.* (1999) use tf-idf to retrieve signature words, NER to retrieve tokens, shallow discourse analysis for cohesion and also use synonym and morphological variants of lexical terms using WordNet. Lin (1999) uses a rich set of features for the creation of feature vector like *Title, Tf & Tf-Idf scores, Position score, Query Signature, IR Signature, Sentence Length, Average Lexical Connectivity, Numerical Data, Proper Name, Pronoun & Adjective, Weekday & Month, Quotation, First Sentence etc*. and use decision tree to learn the feature weights. There are other works based on HMM (Conroy *et al.*, 2008), RTS (Marcu, 1998), lexical chain and cohesion (Barzilay *et al.*, 1997).

3.6 SUBJECTIVITY ANALYSIS

Yu *et al.* (2003) propose to find subjective sentences using lexical resources where the authors hypothesize that subjective sentences will be more similar to opinion sentences than to factual sentences. As a measure of similarity between two sentences they used different measures including shared words, phrases and the WordNet. Potthast *et al.* (2010) focus on extracting top sentiment keywords which is based on Pointwise Mutual Information (PMI) measure (Turney, 2002).

The pioneering work for subjectivity detection is done in (Pang *et al.*, 2004), where the authors use min-cut to leverage the *coherency* between the sentences. The fundamental assumption is that local proximity preserves the objectivity or subjectivity relation in the

review. But the work is completely supervised requiring two levels of tagging. Firstly, there is tagging at the sentence level to train the classifier about the subjectivity or objectivity of individual sentences. Secondly, there is tagging at the document level to train another classifier to distinguish between positive and negative reviews. Hence, this requires a lot of manual effort. Alekh *et al.* (2005) integrate graph-cut with linguistic knowledge in the form of WordNet to exploit similarity in the set of documents to be classified.

3.7 CONCEPT EXPANSION USING WIKIPEDIA

Wikipedia is used in a number of works for concept expansion in IR, for expanding the query signature (Muller *et al.*, 2009; Wu *et al.*, 2008; Milne *et al.*, 2007) as well as topic driven multi document summarization (Wang *et al.*, 2010).

There has been a few works in sentiment analysis using Wikipedia (Gabrilovich *et al.*, 2006; Wang *et al.*, 2008). Gabrilovich *et al.* (2006) focus on concept expansion using Wikipedia where they expand the feature vector constructed from a movie review with related concepts from the Wikipedia. This increases accuracy as it helps in unknown concept classification due to expansion but it *does not* address the concern of separating subjective concepts from objective ones.

These works do not take advantage of the Ontological and Sectional arrangement of the Wikipedia articles into categories. Each Wikipedia movie article has sections like *Plot*, *Cast, Production etc.* which can be explicitly used to train a system about the different aspects of a movie. In this work, our objective is to develop a system that classifies the *opinionated extractive summary* of the movie, requiring no labeled data for training; where the summary is created based on the extracted information from Wikipedia.

3.8 VIDEO CATEGORIZATION FROM TEXT

The video categorization works can be broadly divided under 3 umbrellas: 1. Works that deal with low level features by processing the video frames like the audio, video signals, colors, textures *etc*. 2. Works that deal with textual features like the title, tag, video description, user comments *etc*. and 3. Works that combine low-level features with the textual features. In this section, we discuss only those works that include text as one of the features.

Filippova *et al.*, 2011 show that a text-based classifier, trained on imperfect predictions of weakly supervised video content-based classifiers, outperforms each of them taken independently. They use features from the video title, description, user comments as well as the uploader assigned tags and use a maximum entropy model for classification.

Wang *et al.*, 2010 consider features from the text as well as low-level video features, and propose a fusion framework in which these data sources are combined with the small manually-labeled feature set independently. They use a Conditional Random Field (CRF) based fusion strategy and a Tree-DRF for classification.

Wu *et al.*, 2012 perform query expansion by using contextual information from the web like the related videos and user videos, in addition to the textual features and use SVM in the final phase for classification.

Cui *et al.*, 2010 utilize the content features extracted from the training data to enrich the text based semantic kernels to yield content-enriched semantic kernel which is used in the SVM classifier.

Song *et al.*, 2009 developed an effective semantic feature space to represent web videos consisting of concepts with small semantic gap and high distinguishing ability where Wikipedia is used to diffuse the concept correlations in this space. They use Support Vector Machines with fixed number of support vectors (n-ISVM) for classification.

Huang *et al.*, 2010 also uses different text features and classifiers like the Naïve Bayes, Decision Trees and SVM's for classification.

Yang *et al.*, 2007 propose a semantic modality that includes concept histogram, visual word vector model and visual word Latent Semantic Analysis (LSA); and a text modality that includes the titles, descriptions and tags of web videos. They use various classifiers such as Support Vector Machine (SVM), Gaussian Mixture Model (GMM) and Manifold Ranking (MR) for classification.

Zhang *et al.*, 2006 developed a video categorization framework that combined multiple classifiers based on normal text features as well as users' querying history and clicking logs. They used Naïve Bayes with mixture of multinomials, Maximum Entropy, and Support Vector Machines for video categorization.

Borth *et al.*, 2009 combine the results of different modalities like the uploader generated tags and visual features which are combined using a weighted sum fusion, where SVM's are used with bag of words as features. These categories are refined further by deep-level clustering using probabilistic latent semantic analysis.

All the above works build on supervised classification systems, requiring labeled data for training, mostly using the Support Vector Machines. In this work, we propose a completely unsupervised system, requiring no labeled data, which is the primary difference of our work with those surveyed. Also, the usefulness of Wikipedia and WordNet for concept expansion has not been probed much in earlier video categorization tasks, save a few.

Chapter 4

4. SENTIMENT ANALYSIS IN TWITTER WITH LIGHTWEIGHT DISCOURSE ANALYSIS

This chapter analyzes the effect of discourse coherent relations in sentiment analysis and proposes a lightweight computation model to incorporate discourse features in a bag-of-words model. The roadmap for the rest of the chapter is as follows: *Section 4.1* presents a comprehensive view of the different discourse relations. *Section 4.2* studies the effect of these relations on sentiment analysis and identifies the critical ones. We develop techniques for using the discourse relations to create feature vectors in *Section 4.3*. Lexicon based classification and supervised classification systems are presented in *Section 4.4* to classify the feature vectors. Experimental results are presented in *Section 4.5*, where we use *three* different datasets, from the *Twitter* and *Travel review* domain, to validate our claim. The results are discussed in *Section 4.6*, followed by conclusions.

4.1 CATEGORIZATION OF DISCOURSE RELATIONS

"An important component of language comprehension in most natural language contexts involves connecting clauses and phrases together in order to establish a coherent discourse" (Wolf *et al.*, 2004). A coherently structured discourse is a collection of sentences having some relation with each other. A coherent relation reflects how different discourse segments interact (Hobbs 1985; Marcu 2000; Webber *et al.,* 1999). Discourse segments are non-overlapping spans of text. The exact definition of a discourse segment is a matter of debate. The interaction relations between discourse segments may be of various forms as listed in *Table 4.1*.

Our work, almost entirely, rests on this platform, where we identify the relations from *Table 4.1*, which can affect the analysis of opinions most in a discourse segment. *Table 4.2* provides some examples, taken from *Twitter*, to illustrate the effect of conjunctions in discourse analysis. These examples are similar to the ones in Polanyi *et al.* (2004) and Taboada *et al.* (2008). In *Table 4.2*, the words *in bold* connect the discourse segments in brackets.

Coherence Relations	Conjunctions
Cause-effect	because; and so
Violated Expectations	although; but; while
Condition	if…(then); as long as; while
Similarity	and; (and) similarly
Contrast	by contrast; but
Temporal Sequence	(and) then; first, second, … before; after; while
Attribution	according to …; …said; claim that …; maintain that …; stated
Example	for example; for instance
Elaboration	also; furthermore; in addition; note (furthermore) that; (for,in,with) which; who; (for,in,on, against, with) whom
Generalization	in general

Table 4.1: Contentful Conjunctions used to Illustrate Coherence Relations (Wolf *et al.*, 2005)

4.2 DISCOURSE RELATIONS CRITICAL FOR SENTIMENT ANALYSIS (SA)

Not all of the discourse relations are significant from the point of view of sentiment analysis. This section examines the role of the critical ones in SA.

4.2.1 Violated Expectations and Contrast

A simple bag-of-words model will classify *Example 2* (*Table 4.2*) as neutral. This is because it has one positive term *excited* and one negative phrase *not really liking*. However, it represents a positive emotion of the opinion holder, due to the segment after the connective *despite*. In *Example 5*, *brightened* is positive and *poorly* is negative. Hence the overall polarity is un-decided. But it should have been *positive*, since the segment following *but* gives the overall impression of the opinion-holder, which is positive.

Violating expectation conjunctions oppose or refute the neighboring discourse segment. We further categorize them into the following *two* sub-categories: *Conj_Fol* and *Conj_Prev.*

Conj_Fol is the set of conjunctions that give more importance to the discourse segment that follows them. *Conj_Prev* is the set of conjunctions that give more importance to the previous discourse segment.

Thus, in *Example 5* of *Table 4.2*, the discourse segment following *but* should be given more weight. In *Example 2*, the discourse segment preceding *despite* should be given more weight.

1. Cause-effect: *(YES! I hope she goes with Chris)* **so** *(I can freak out like I did with Emmy Awards.)*
2. Violated Expectations: *(i'm quite excited about Tintin),* **despite** *(not really liking original comics.)*
3. Condition: **If** *(MicroMax improved its battery life), (it wud hv been a gr8 product).*
4. Similarity: *(I lyk Nokia)* **and** *(Samsung as well).*
5. Contrast: *(my daughter is off school very poorly),* **but** *(brightened up when we saw you on gmtv today).*
6. Temporal Sequence: *(The film got boring)* **after a while**.
7. Attribution: *(Parliament is a sausage-machine: the world)* **according to** *(Kenneth Clarke).*
8. Example: *(Dhoni made so many mistakes…)* **for instance***, (he shud've let Ishant bowl wn he was peaking).*
9. Elaboration: **In addition** *(to the worthless direction), (the story lacked depth too).*
10. Generalization: **In general***,(movies made under the RGV banner) (are not worth a penny).*

Table 4.2: Examples of Discourse Coherent Relations

4.2.2 Conclusive or Inferential Conjunctions

These are the set of conjunctions, *Conj_infer*, that tend to draw a conclusion or inference. Hence, the discourse segment following them (*subsequently* in *Example 14*) should be given more weight.

Example 14: @User *I was nt much satisfied with ur so-called gud phone and* **subsequently** *decided to reject it.*

4.2.3 Conditionals

In *Example 3*, both *improve* and *gr8* represent a *high degree* of positive sentiment. But the presence of *if* tones down the final polarity as it introduces a hypothetical situation in the context. The *if…then…else* constructs depict situations which may or may not happen subject to certain conditions.

In our work, the polarity of the discourse segment in a conditional statement is toned down, in *lexicon-based classification*. In *supervised classifiers*, the conditionals are marked as

features. Such statements are not completely ignored as objective, as they bear some sentiment polarity.

4.2.4 Modals

Events that have happened, events that are happening or events that are certain to occur are called *realis events.* Events that have possibly occurred or have some probability to occur in the distant future are called *irrealis events.* Thus, it is important to distinguish between *real* situations and *hypothetical* ones. The modals (*might, may, could, should, would etc*.) depict *irrealis events. Example 3* does not necessarily talk of MicroMax being *great,* but talks of its *possibility* of being *great* subject to certain conditions (its *battery life*). These constructs cannot be handled by taking a simple majority valence of terms.

We further divide the modals into *two* sub-categories: *Strong_Mod* and *Weak_Mod*.

Strong_Mod is the set of modals that express a higher degree of uncertainty in any situation.

Weak_Mod is the set of modals that express lesser degree of uncertainty and more emphasis on certain events or situations.

Example 15 (Strong Modals): *Unless I missed the announcement their God is now featured on postage stamps, it* **might** *be a hard sell.*
He **may** *be a rising star.*

Example 16 (Weak Modals): *G.E 12* **must** *be the most deadly General Election for politicians ever.*
Our civil service **should** *work without TD interference.*

In our work, the polarity of the discourse segment neighboring a *strong modal* is toned down in *lexicon-based classification,* similar to the *conditionals*, as it expresses a higher degree of uncertainty. In *supervised classifiers*, the modals are marked as features.

4.2.5 Negation

The negation operator (Example: *not, neither, nor, nothing etc*.) inverts the sentiment of the word following it. The usual way of handling negation in SA is to consider a window of size *n* (typically 3-5) and reverse the polarity of all the words in the window.
Example 17 (Negation): *I do not like Nokia but I like Samsung.*

Here negating all the words in a window of size 5 reverses the polarity of "*like*" for Samsung as well; this is undesirable.

We consider a negation window of size 5 and reverse all the words in the window, till either the window size exceeds or a *violating expectation* (*or a contrast*) conjunction is encountered. Hence, the scope of reversing polarity is limited to the appearance of *but* in the above example.

4.2.6 Other Discourse Relations

Sentences under *Cause-Effect, Similarity, Temporal Sequence, Attribution, Example, Generalization and Elaboration,* provide no contrasting, conflicting or hypothetical information. They can be handled by taking a simple majority valence of the individual terms, as in a bag-of-words model.

Table 4.3 lists all the essential discourse relations discussed in this section. The relations have been compiled from Hobbs (1985), Polanyi *et al.* (2004) and Taboada *et al.* (2008).

Discourse Relations	Attributes
Conj_Fol	*but, however, nevertheless, otherwise, yet, still, nonetheless*
Conj_Prev	*till, until, despite, in spite, though, although*
Conj_Infer	*therefore, furthermore, consequently, thus, as a result, subsequently, eventually, hence*
Conditionals	*If*
Strong_Mod	*might, could, can, would, may*
Weak_Mod	*should, ought to, need not, shall, will, must*
Neg	*not, neither, never, no, nor*

Table 4.3: List of Discourse Coherent Features

4.3 ALGORITHM TO HARNESS DISCOURSE INFORMATION

The discourse relations (identified in *Section 4*) are used to create a feature vector, according to *Algorithm 4.1*.

In *Step 1*, we mark all the *conditionals* and *strong modals* which are handled separately by the lexicon-based classifier and the supervised classifier.

In *Step 2* and *Step 3*, the weight of any word appearing before *Conj_Prev* and after *Conj_Fol* or *Conj_Infer* is incremented by 1.

In *Step 4*, the polarity of all the words appearing within a window (*Neg_Window* is taken as 5), from the occurrence of a negation operator and before the occurrence of a *violating expectation* conjunction, are reversed.

Finally, we get the feature vector $\{w_{ij}, f_{ij}, flip_{ij}$ and $hyp_{ij}\}$ for all the words in the review.

Here, the assumption is that the effect of any conjunction is restricted to continuous spans of text till another conjunction or the sentence boundary.

4.4 FEATURE VECTOR CLASSIFICATION

We devised a lexicon based system as well as a supervised system for feature vector classification.

Let a user post *R* consist of '*m*' sentences s_i *(i=1…m)*, where each s_i consist of n_i words w_{ij} *(i=1…m, j=1…n_i)*. Let f_{ij} be the weight of the word w_{ij} in sentence s_i, initialized to 1. Let *A* be the set of all discourse relations in *Table 4.3*. Let $flip_{ij}$ be a variable which indicates whether the polarity of w_{ij} should be flipped or not. Let hyp_{ij} be a variable which indicates the presence of a *conditional or a strong modal* in s_i.

for *i=1…m*
  for *j=1…n_i*
    f_{ij}=1;
  hyp_{ij}=0;
1. **if** w_{ij} ∈ Conditionals *or* w_{ij} ∈ Strong_Mod
    hyp_{ij}=1;
  end if
end for
  for *j=1…n_i*
    $flip_{ij}$=1;
2. **if** w_{ij} ∈ Conj_Fol *or* w_{ij} ∈ Conj_Infer
      for *k=j+1…n_i and* $w_{ij} \notin A$
        f_{ik}+=1;
    end for
  end if
3. **else if** w_{ij} ∈ Conj_Prev
      for *k=1…j-1 && $w_{ij} \notin A$*
        f_{ik}+=1;

end for

end if

4. **else if** $w_{ij} \in Neg$

for *k=1…Neg_Window and*

$w_{ik} \notin Conj\_Prev$ *and* $w_{ik} \notin Conj\_Fol$

$flip_{i,j+k}$=-1;

end for

end if

end for

Input: Review *R*

Output: $w_{ij}, f_{ij}, flip_{ij}, hyp_{ij}$

Algorithm 4.1: Using the Discourse Relations to Create the Feature Vector

4.4.1 Lexicon based Classification

The Bing Liu sentiment lexicon (Hu *et al.*, 2004) is used to find the polarity $pol(w_{ij})$ of a word w_{ij}. It contains around 6800 words which are manually polarity labeled. The polarity of the review (*pos* or *neg*) is given by,

$$sign(\sum_{i=1}^{m}\sum_{j=1}^{n_i} f_{ij} \times flip_{ij} \times p(w_{ij}))$$

$$where\ p(w_{ij}) = pol(w_{ij})\ \ if\ hyp_{ij} = 0$$

$$= \frac{pol(w_{ij})}{2} \quad if\ hyp_{ij} = 1$$

Equation 4.1: Polarity Classification of a Review

Equation 4.1 finds the weighted, signed polarity of a review. The polarity of each word, $pol(w_{ij})$ *being +1 or -1*, is multiplied with its discourse-weight f_{ij} (*Algorithm 1*), and all the weighted polarities are added. $Flip_{ij}$ indicates if the polarity of w_{ij} is to be negated.

In case there is any *conditional* or *strong modal* in the sentence (indicated by $hyp_{ij} = 1$), then the polarity of every word in the sentence is toned down, by considering half of its assigned polarity ($\frac{+1}{2}$ *or* $\frac{-1}{2}$).

Thus, if *good* occurs in the user post twice, it will contribute a polarity of $+1 \times 2 = +2$ to the overall review polarity,$if\ hyp_{ij} = 0$. In the presence of a *strong modal* or *conditional*, it will contribute a polarity of $\frac{+1}{2} \times 2 = +1$.

All the *stop words*, *discourse connectives* and *modals* are ignored during the classification phase, as they have a zero polarity in the lexicon.

4.4.2 Supervised Classification

The Support Vector Machines have been found to outperform other classifiers, like *Naïve Bayes* and *Maximum Entropy*, in sentiment classification (Pang *et al.*, 2002). Hence, in our work, SVM's are used to classify the set of feature vectors $\{flip_{ij}, w_{ij}, f_{ij}$ and $hyp_{ij}\}$.

Features used in the Support Vector Machines:

N-grams – *Unigrams* along with *Bigrams* are discarded.

Stop Words – All the stop words (like *a, an, the, is, etc*.) and discourse connectives are ignored.

Feature Weight – In the *baseline bag-of-words* model, the feature weight has been taken as the feature frequency *i.e.* the number of times the unigram or bigram appears in the text. In the *discourse-based bag-of-words* model, the *discourse-weighted frequency* of a word is considered. *Algorithm 1* assigns a weight f_{ij} to every occurrence of a word w_{ij} in the post. If the same word occurs multiple times, the weights from its multiple occurrences will be added and used as a feature weight for the word.

Modal and Conditional Indicator – This is a Boolean variable which indicates the presence of a strong modal or conditional in the sentence (*i.e.* $hyp_{ij}=1$).

Stemming – All the words are stemmed in the text so that "*acting*" and "*action*" have a single entry corresponding to "*act*".

Negation – A Boolean variable ($flip_{ij}$) is appended to each word (w_{ij}) to indicate whether it is negated or not (*i.e.* $flip_{ij}=1$ or $flip_{ij}=0$).

Emoticons – An emoticon dictionary[6] is used to map each emoticon to a *positive* or *negative* class. Subsequently, the emoticon class information is used in place of the emoticon.

Part-of-Speech Information – The part-of-speech information is also used with a word.

Feature Space - In the *lexeme feature space* individual words are used as features; whereas in the *sense space,* the sense of the word (synset-id) is used in place of the word. A synset is a

[6] http://chat.reichards.net/smiley.shtml

set of synonyms that collectively disambiguate each other to give a unique sense to the set, identifiable by the *synset-id*. This helps to distinguish between the various senses of a word.

For example, the word *bank* has 18 senses (10 Noun senses and 8 Verb Senses). Consider the two senses of a *bank* – 1) *Bank* in the sense of "*a financial institution*", identifiable by the synset "*depository financial institution, bank, banking concern, banking company",* and 2) *Bank* in the sense of *relying,* identifiable by the synset "*trust, swear, rely, bank"*. Now, the first sense has an objective polarity whereas the second sense has a positive polarity. This distinction cannot be made in the lexeme feature space, where we consider only the first sense of a word. In this work, the *micro-blog* data is evaluated in the *lexeme* feature space, whereas the *travel review* data is evaluated in both the *lexeme* space and the *sense* space.

Algorithm 1 outputs feature vectors for the lexeme feature space. An automatic word sense disambiguation algorithm, *Iterative Word Sense Disambiguation* (Khapra *et al.*, 2010), is used to transform each word (in the user post) in the lexeme space to the corresponding synset_id in the sense space. *Algorithm 1* is then used on the transformed user post to find the feature weights, similarly, as in the lexeme space.

4.5 EVALUATION

4.5.1 Dataset

Dataset 1: Twitter is crawled using a Twitter API[7] and 8507 tweets are collected based on a total of around 2000 different entities from 20 different domains. The following domains are used for crawling data: *Movie, Restaurant, Television, Politics, Sports, Education, Philosophy, Travel, Books Technology, Banking & Finance, Business, Music, Environment, Computers, Automobiles, Cosmetics brands, Amusement parks* and *Eatables and History.* These are manually annotated by 4 annotators into four classes - *positive, negative, objective-not-spam* and *objective-spam*. The objective-not-spam category contains tweets which are objective in nature but are not spams. The objective-spam category contains spam tweets which are subsequently ignored during evaluation.

Dataset 2: Following the works of Read *et al.* (2005), Alec *et al.* (2009), Pak *et al.* (2010) and Gonzalez *et al.* (2011) we create an artificial dataset using *hashtags*. The Twitter API is used to collect another set of 15,214 tweets based on *hashtags*. Hashtags *#positive, #joy, #excited,*

[7] http://search.Twitter.com/search.atom

#happy etc. are used to collect tweets bearing positive sentiment, whereas hashtags like *#negative, #sad, #depressed, #gloomy, #disappointed etc*. are used to collect negative sentiment tweets. *Hashtag keywords are subsequently removed from the tweets.*

Dataset 3: *Travel Review* Dataset in Balamurali *et al.* (2011) contains 595 polarity-tagged documents for each of the positive and negative classes. All the words in the travel review documents are automatically tagged with their corresponding synset-id's using *Iterative Word Sense Disambiguation* algorithm (Khapra *et al.*, 2010).

Manually Annotated Dataset 1				
#Positive	#Negative	#Objective Not Spam	#Objective Spam	Total
2548	1209	2757	1993	8507
Auto Annotated Dataset 2				
#Positive		#Negative		Total
7348		7866		15214

Table 4.4: Dataset 1 and 2 Statistics

4.5.2 Evaluation on the Twitter Dataset 1 and 2

The crawled tweets are pre-processed before evaluation. All the *links* (urls) in the tweets are replaced by *#link*. All the *user id's* in the tweets are replaced by *#user*.

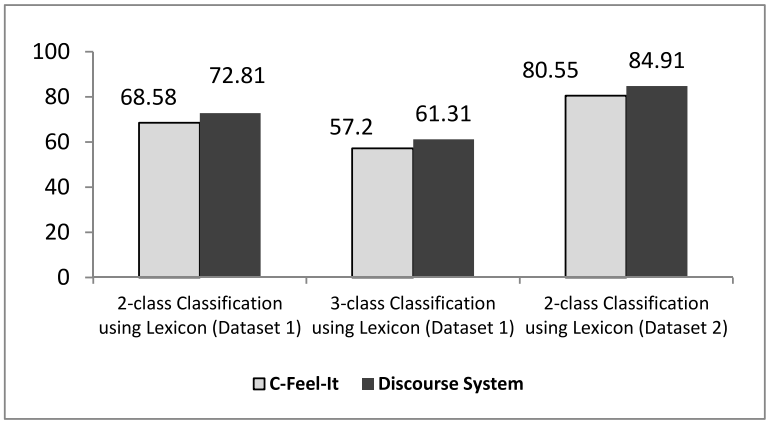
Evaluations are performed in *Dataset 1* and *2* under a *2-class* and a *3-class* classification setting. In the 2-class setting, only *positive* and *negative* tweets are considered; whereas in the 3-class setting *positive*, *negative* and *objective-not-spam* tweets are considered. All the experiments in these two datasets are performed in the **lexeme feature space** using *lexicon-based classification* as well as *supervised classification*.

The baseline system, for this part of the evaluation, is taken as *C-Feel-It* (Joshi *et al.*, 2011). It is a rule-based system which implements a bag-of-words model using lexicon-based classification. The accuracy comparisons between C-Feel-It and the discourse system are performed under identical settings. The only difference between the two systems is the handling of *connectives, modals, conditionals* and *negation*, as indicated by *Algorithm 1*.

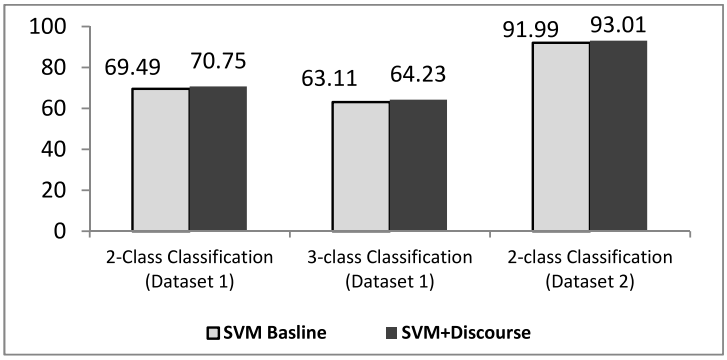
Graph 4.1 shows the accuracy comparison between C-Feel-It and the discourse system using lexicon-based classification, in Datasets 1 and 2, under a 2-class and a 3-class setting.

Graph 4.2 shows the accuracy comparison between the baseline SVM and SVM integrated with discourse features, in Datasets 1 and 2, under a 2-class and a 3-class setting. All the SVM features discussed in *Section 6.2,* except the discourse features arising out of the

incorporation of *discourse* weighting, *modal and conditional indicator* and *negation*, are used in the baseline SVM model. A linear kernel, with default parameters, is used in the SVM[8] with 10-fold cross-validation.



Graph 4.1: Accuracy Comparison between C-Feel-It and Discourse System using Lexicon in Datasets 1 and 2



Graph 4.2: Accuracy Comparison between Baseline SVM and SVM with Discourse in Datasets 1 and 2

4.5.3 Evaluation of the Travel Review Dataset 3

The *travel review* dataset in Balamurali *et al.* (2011) is used to determine whether our discourse-based approach performs well for structured text as well. This evaluation is done under a 2-class classification setting in the *lexeme space* as well as the *sense space*.

Table 4.5 shows the accuracy comparison between the baseline bag-of-words model and bag-of-words model integrated with discourse features using lexicon-based classification, in Dataset 3, under a 2-class setting.

[8] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

Sentiment Evaluation Criterion	Accuracy
Baseline Bag-of-Words Model	69.62
Bag-of-Words Model + Discourse	**71.78**

Table 4.5: Accuracy Comparison between Bag-of-Words and Discourse System using Lexicon in Dataset 3

An automatic word sense disambiguation algorithm, *Iterative Word Sense Disambiguation* (Khapra *et al.*, 2010), has been used to auto-annotate the words in the review with their corresponding synset-id's. The same dataset is used in this work. *Table 4.6*, adapted from Balamurali *et al.* (2011), shows the performance of the auto-annotation algorithm IWSD in the dataset.

POS	#Words	P(%)	R(%)	F-Score(%)
Noun	12693	75.54	75.12	75.33
Adverb	4114	71.16	70.9	71.03
Adjective	6194	67.26	66.31	66.78
Verb	11507	68.28	67.97	68.12

Table 4.6: IWSD Annotation Statistics (*P-Precision, R-Recall)*

A linear kernel, with default parameters, is used in the SVM[4] with 5-fold cross-validation. *Table 4.7* shows the performance of the discourse system along with the compared system using different features, on Dataset 3, using supervised classification. The features used in the SVM, for this part of the evaluation, include *stop word removal*, *no stemming*, *part-of-speech information* and *unigrams*. These features are used in all the systems in *Table 4.7,* including the discourse one. *Table 4.7* shows the system accuracy under four scenarios:

1. When only unigrams are used
2. When only sense of unigrams are used
3. When unigrams are used along with their senses (synset-id's)
4. When unigrams are used with senses and discourse features

Systems	Accuracy (%)
Baseline Accuracy (Only Unigrams)	84.90
Balamurali *et al.*, 2011 (Only IWSD Sense of Unigrams)	85.48
Balamurali *et al.*, 2011 (Unigrams+IWSD Sense of Unigrams)	86.08
Unigrams + IWSD Sense of Unigrams+Discourse Features	**88.13**

Table 4.7: Accuracy Comparisons in Travel Review Dataset 3

4.6 DISCUSSIONS

Accuracy improvements over the baseline and the compared systems in all the datasets, under different settings and feature space, clearly signify the effectiveness of incorporating discourse information for sentiment classification. The bag-of-words model integrated *with discourse information* outperforms the bag-of-words model, *without this information*, under all the settings; although, the performance improvements vary in different settings. Statistical tests have been performed and all the accuracy improvements have been found to be statistically significant with *99%* level of confidence.

4.6.1 Accuracy Comparison between C-Feel-It and Discourse System

These comparisons are performed under a 2-class and a 3-class classification setting, using lexicon-based classification, in the lexeme space under identical settings - the only difference being the incorporation of discourse features. In *Dataset 1*, there is an accuracy improvement of around *4%* over C-Feel-It for both 2-class and 3-class classification. The discourse system accuracy at *72.81%* for 2-class classification is higher than that of the 3-class classification accuracy of *61.31%*. This shows that 3-class classification of tweets is much more difficult than 2-class classification.

4.6.2 Accuracy Comparison between Baseline SVM and Discourse System

These comparisons are performed under a 2-class and a 3-class classification setting, using supervised classification, in the lexeme space. A similar feature set, except the discourse features, is used for both the systems.

In *Dataset 1*, there is an accuracy improvement of *1%* in both the 2-class and 3-class classification, which has been found to be statistically significant. In *Dataset 2*, there is an accuracy improvement of *2%* over baseline SVM for 2-class classification.

It is observed that in the 2-class setting, the discourse system performs better in the lexicon-based classification with an accuracy of *72.81%* compared to the supervised classification accuracy of *70.75%*. This is contrary to the common scenario in text classification, where the supervised classification system always performs much better than the lexicon-based classification. This may be due to the very sparse feature space, owing to the length limit of tweets (140 characters).

4.6.3 Effect of Artificially Created Training Data

There has been a lot of work in Twitter that collect data based on specific features like *hashtags* (Alec *et al.*, 2009; Read *et al.*, 2005; Pak *et al.*, 2010)*, emoticons* (Gonzalez *et al.*, 2011) *etc*. and auto-annotate the tweets based on these features. Although these systems achieve a very high accuracy, they remain biased towards these special features. In this work, we showed that although a system may work very well on a dataset based on a specialized feature set with hashtags (*Dataset 2*), it does not necessarily work well in a general setting (*Dataset 1*). This is evident in the performance of the discourse system in *Dataset 2* (created based on *hashtags*) where it attains a high accuracy of *84.91%* compared to the accuracy of *72.81%* in *Dataset 1* (manually annotated general purpose data) for lexicon-based classification. In supervised classification, the discourse system has an accuracy of *70.75%* in *Dataset 1* and *93.01%* in *Dataset 2*. This shows that the specialized set of features used to crawl the data actually gives away the sentiment *explicitly*, unlike in the general dataset which may have latent sentiment based out of *sarcasm, jokes, teasers* and other *implicit* sentiment, which is quite difficult to detect.

4.6.4 Accuracy Comparison in Dataset 3

In the *Travel review* dataset, lexicon-based classification yielded an accuracy improvement of *2%* for the discourse model over simple bag-of-words model, in the *lexeme space*.

In the SVM classification, in the *sense space,* under a 2-class setting, the discourse system achieved an accuracy of *88%* compared to *86%* accuracy of Balamurali *et al.* (2011). A similar feature set has been used in both the models, which indicates that the performance improvement is due to the incorporation of discourse features in SVM.

4.6.5 Drawbacks

The lexicon-based classification suffers from the usage of a generic lexicon in the *lexeme space*, where it cannot distinguish between the various senses of a word. The lexicons do not have entries for the interjections like *wow*, *duh etc*. which are strong indicators of sentiment. The frequent spelling mistakes, abbreviations and slangs used in the tweets do not have entry in the lexicons. For example, *love* and *great* are frequently written as *luv* and *gr8*, which will not be detected. A spell-checker may help the system in this regard.

The supervised system suffers from a sparse feature space due to very short contexts. A concept expansion approach, to expand the feature vectors, may prove to be useful. This is due to the extensive world knowledge embedded in the tweets. For example, the tweet "*He is*

a Frankenstein" is tagged as objective. The knowledge that *Frankenstein* is a negative concept is not present in the lexicon. The IWSD algorithm for automatic sense annotation has an F-Score of *70%*, which means many of the word-senses were wrongly tagged. In case a better WSD algorithm is used, higher system accuracy can be achieved in the travel dataset.

In the absence of *parsing* and *tagging* information, the scope of the discourse marker has been heuristically taken till the sentence boundary or till the next discourse marker. Consider the sentence, "*I wanted to follow my dreams and ambitions despite all the obstacles, but I did not succeed.*" Here *want* and *ambition* will get the polarity *+2 each*, as they appear before *despite*; *obstacle* will get a polarity *-1* and *not succeed* will get a polarity *-2*. Thus the overall polarity is *+1,* whereas the overall sentiment should be *negative*. This is because we do not consider the *positional importance* of a discourse marker in the sentence and consider all the discourse markers to be equally important. A better method is to give a ranking to the discourse markers based on their *positional* and *pragmatic* importance.

4.7 SUMMARY

We showed that the incorporation of discourse markers in a bag-of-words model improves the sentiment classification accuracy by *2 - 4%*. This approach is particularly beneficial for- 1) applications dealing with noisy text where *parsing* and *tagging* do not perform very well, and 2) applications, requiring a fast response time, where employing a heavy linguistic tool like *parsing* will be detrimental to its performance due to the increased processing time.

Most of the works in micro-blogs, like *Twitter*, build on a bag-of-words model that ignores the discourse markers. We demonstrated an approach to incorporate discourse information to improve their performance, retaining the simplicity of the bag-of-words model. We validated this claim on two different datasets (manually and automatically annotated) from *Twitter*, where we achieved an accuracy improvement of *4%* for lexicon-based classification over an existing application (Joshi *et al.,* 2011), and 2% for supervised classification over the baseline SVM with advanced features.

We also showed that our method fares well for structured reviews as well, where we achieved similar accuracy improvements over a state-of-the-art system (Balamurali *et al.*, 2011).

Chapter 5

5. FEATURE SPECIFIC SENTIMENT ANALYSIS OF PRODUCT REVIEWS

This chapter analyzes the association between words forming opinion expressions, using dependency parsing, and proposes a computational model to incorporate feature specificity in a sentiment analysis system. The roadmap the remaining part of this chapter is as follows: *Section 5.1* gives the problem statement along with the challenges associated with it. *Section 5.2* gives the algorithm to extract features and their associated opinion expressions. It presents a graph based representation of the features and their relations, which is partitioned to obtain feature specific opinions. *Section 5.3* presents a rule-based and supervised classification system to find the sentiment polarity. We present the learning of the domain independent parameters in *Section 5.4*, followed by extensive experiments across various product domains in review blogs and Twitter to validate our claims in *Section 5.5*.

5.1 PROBLEM STATEMENT

Given a product review containing multiple features and varied opinions, the objective is to extract expressions of opinion related to a *target feature* and classify it as positive or negative. The objectives can be summarized is:

1. Extract all the features from the given review

   In the absence of any prior information about the domain of the review (in the form of untagged or tagged data belonging to that domain), this will give a list of *potential features* in that review which needs to be pruned to obtain the exact features.

2. Extract opinion words referring to the target feature

   The opinion words are not only Adjectives like *hate, love* but also consist of other POS categories like Nouns (*terrorism)*, Verbs (*terrify)* and Adverbs (*gratefully*). A naïve method, like extracting the opinion words closest to the *target feature*, does not work so well when the sentence has multiple features and distributed emotions.

3. Classify the extracted opinion words as positive or negative

5.2 FEATURE SPECIFIC SENTIMENT ANALYSIS

In this section, we will first outline a method to extract features and their associated relations.

5.2.1 Feature Extraction

We will elaborate 2 methods for extracting features corresponding to the availability of domain knowledge.

5.2.1.1 Feature Extraction in Absence of Domain Knowledge

In the absence of any prior information about the product domain, we can make a list of *potential features* in the review by constraining the features only to be Nouns (like *multimedia, firmware, display, color etc*.). All the words in the sentence are POS-tagged and all the Nouns are retrieved. Initially, all the Nouns are treated as features and added to the *feature list* ***F***.

Consider the review,

"*I have an ipod and it is a great buy but I'm probably the only person that dislikes the iTunes software."*

F = {*ipod, buy, person, software*}

This forms our initial feature set. But the intended features are *ipod* and *software.* We will later present an algorithm to prune this initial feature set, such that any 2 features **strongly related** will be merged. Thus, *buy* will be merged with *ipod* when the target feature is *ipod,* and {*person, software*} will be pruned. If the target feature is *software*, *person* will be merged with *software,* and {*ipod, buy*} will be pruned.

5.2.1.2 Feature Extraction in Presence of Domain Knowledge

If domain information is available (in the form of crawled reviews from the domain in focus, when the product domain has been identified) we can extract all the features in the domain using *Latent Dirichlet Allocation* (Blei *et al.,* 2003) or *HMM-LDA* (Griffiths *et al.,* 2005). This can be used to directly prune the feature set ***F*** to obtain the exact set of features.

5.2.2 Relation Extraction

We identify 2 kinds of relations between the words in a sentence that associate them to form a coherent review:

1. Direct Neighbor Relation

Let *Stopwords* be the list of pre-compiled stop words occurring frequently in any text. This comprises mainly of *be* verbs, personal pronouns, prepositions, conjunctions *etc*. All Nouns, Adjectives, Adverbs, Verbs (except *be* verbs) are excluded from the list.

Consider a sentence *S* and 2 consecutive words $w_i, w_{i+1} \in S$. *If* $w_i, w_{i+1} \notin Stopwords$, then they are directly related. This helps us to capture *short range syntactic dependencies*.

2. Dependency Relation

Let *Dependency_Relation* be the list of significant relations. We call any *dependency relation* significant, if

- It involves any *subject*, *object* or *agent* like *nsubj, dobj, agent etc*
- It involves any *modifier* like *advmod, amod etc*
- It involves *negation* like *neg*
- It involves any *preposition* like *prep_of*
- It involves any *adjectival or clausal* component like *acomp, xcomp*

The above set of relations is not minimal, in the sense that not all of them are equally significant in capturing the semantic coherence in reviews. We will later show how to prune the above set of relations, to obtain a minimal set of significant relations, by a small seed set of data using ablation test.

Any 2 words w_i and w_j in *S* are directly related, *if*

$\exists D_l \;\; s.t. \;\; D_l(w_i, w_j) \in Dependency\_\mathrm{Re}lation$.

This helps us to capture *long range semantic dependencies*.

The direct neighbor and dependency relations are combined to form the master *relation set* ***R***.

We now formulate the following hypothesis:

More closely related words come together to express an opinion about a feature.

If there are *'n'* features of a product in a sentence, then those words that are most closely related (in terms of relations defined above) to a feature *'i'* will come together to express some opinion about it, rather than about some other feature *'j'*, to which they are not so closely associated.

For Example: "*I want to use Samsung which is a great product but am not so sure about using Nokia".*

Here {*great*, *product*} are related by an adjectivial modifier relation, and {*product*, *Samsung*} are related by a relative clause modifier relation. Thus {*great, Samsung*} are transitively related. **Here {*great, product*} are more closely related to Samsung than they are to Nokia.** Thus {*great, product*} come together to express an opinion about the entity "Samsung" than about the entity "Nokia".

5.2.3 Graph Representation

Given a sentence *S*, let ***W*** be the set of all words in the sentence *S*.

A Graph $G(W, E)$ is constructed such that any $w_i, w_j \in W$ are directly connected by $e_k \in E$, *if* $\exists R_l \;\; s.t. \;\; R_l(w_i, w_j) \in R$.

In other words, in the graph *G* all the words in the given sentence are considered as vertices. Any 2 vertices are connected, if there is any relation between them governed by the relation set ***R***.

5.2.4 Dependency Extraction

We have the set of all features *F* and a graph *G*. Let $f_t \in F$ be the target feature. For example in Section 4.1, *ipod* or *software* can be the *target* feature *i.e.* the feature with respect to which we want to evaluate the sentiment of the sentence.

Let there be *'n'* features where *n* is the dimension of *F*. The algorithm for extracting the set of words $w_i \in S$, that express any opinion about the target feature f_t proceeds as follows:

i. Initialize *n* clusters $C_i \;\; \forall \; i = 1 \ldots n$

ii. Make each $f_i \in F$ the clusterhead of C_i. The target feature f_t is the clusterhead of C_t. Initially, each cluster consists only of the clusterhead.

iii. Assign each word $w_j \in S$ to cluster C_k *s.t.*, $k = argmin_{i \in n} dist(w_j, f_i)$
Where $dist(w_j, f_i)$ *gives the number of edges, in the shortest path, connecting* w_j *and* f_i *in G.*

iv. Merge any cluster C_i with C_t *if* $dist(w_j, f_i) < \theta$,
Where θ is some threshold distance.

v. Finally the set of words $w_i \in C_t$ gives the opinion expression regarding the target feature f_t.

Algorithm 5.1: Dependency Parsing Based Clustering for Sentiment Analysis

In words, we initialize '*n*' clusters C_i, corresponding to each feature $f_i \in F$ *s.t.* f_i is the clusterhead of C_i. We assign each word $w_i \in S$ to the cluster whose clusterhead is closest to it. The distance is measured in terms of the number of edges in the shortest path, connecting any word and a clusterhead. Any 2 clusters are merged if the distance between their clusterheads is less than some threshold. Finally, the set of words in the cluster C_t, corresponding to the target feature f_t gives the opinion about f_t.

5.2.5 Feature Clustering with Example

Consider the review, "*I have an ipod and it is a great buy but I'm probably the only person that dislikes the itunes software"*

Here, $\boldsymbol{F}$={*ipod,buy, person and software*}forms our initial feature set (represented by rectangles in *figure 5.1*). The target feature is $\boldsymbol{f_t}$ = *ipod*. The graph consists of all the words in the sentence as vertices. All the words are connected by relations defined by the master relation set $\boldsymbol{R}$ (shown by thin edges in *figure 5.1*). The target cluster $\boldsymbol{C_t}$ has the clusterhead $\boldsymbol{f_t}$. {*I, have, it*} are closest to *ipod* and are assigned to the corresponding cluster whereas {*great, probably, but, im*} are closest to *buy* and assigned to its corresponding cluster (the assignment is shown by bold arrows in *figure 5.1*). Now, *ipod* and *buy* are related through *it*. The intercluster-distance between them is 2 which is less than θ=3 and thus the 2 clusters are merged. So, *buy* with all its members is assigned to the target cluster $\boldsymbol{C_t}$. {*an, is, a*}are ignored as StopWords.

Finally $\boldsymbol{C_t}$ comprises of {*I, have, ipod, it, great,buy, probably, but, im*} which represents the opinion expression about the target feature $\boldsymbol{f_t}$= ipod.

5.3 CLASSIFICATION OF EXTRACTED FEATURES

Now, have the set of opinion words $w_i \in C_t$, that describes the target feature f_t.

5.3.1 Rule Based Classification

We use a sentiment lexicon to find the polarity of each word $w_i \in C_t$. If the number of words tagged positive is greater than that tagged negative, we conclude the sentiment regarding the target feature f_t, to be positive or else negative.

5.3.2 Supervised Classification

Each sentence in the review is represented as a vector consisting of the target feature f_t and its associated opinion words $w_i \in C_t$. These set of vectors are fed into any supervised classification system like the SVM.

5.4 LEARNING PARAMETERS

We have 2 principal parameters to learn, the significant relation set and the merging threshold.

a. Significant Relation Set

Dependency Parsing gives more than 40 relations, not all of which are equally significant. In order to obtain the subset of relations, which are most significant, we have to probe the entire relation space of $O(2^{40})$. It is infeasible to perform an exhaustive search. So, we partition the relation space in 3 parts:

- Relations that *should be included* in *R*

These consist of the relations *nsubj, nsubjpass, dobj, amod, advmod, nn, neg.*

- Relations that *should not be included* in *R*

These consist of relations irrelevant to our purpose like *numeric modifiers*, *abbreviation relations etc*.

- Relations that *may be included* in *R*

This partition consists of around 21 relations which may or may not be significant.

We now perform leave-one-relation out test or *ablation test*. In this, we leave out one relation at a time and compute the overall accuracy of sentiment classification with the remaining relations. We want to find out which of the relations in the 3rd partition causes *significant* accuracy change. We select an arbitrary domain to perform this test and cross-validate in another domain. We used the labeled data from Hu and Liu *et al.* (2004) for learning parameters.

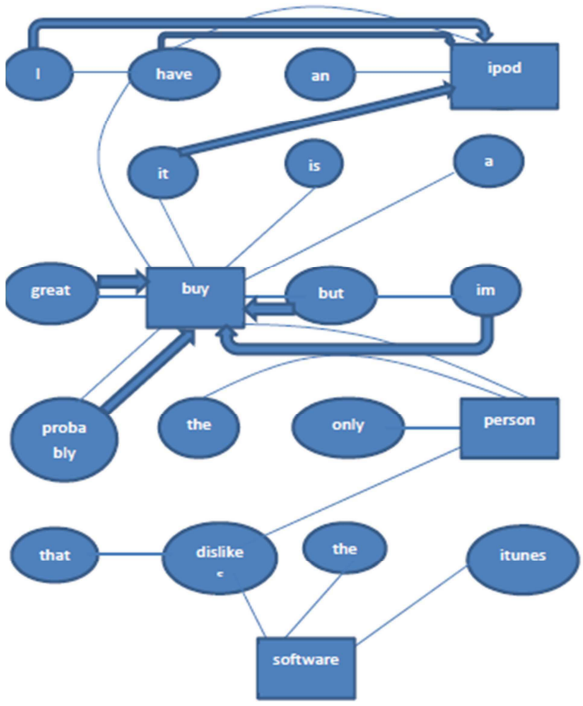


Figure 5.1: Dependency Parsing based Clustering of Features

Relations	Accuracy (%)
All	63.5
Dep	**67.3**
Rcmod	**65.4**
xcomp, conj_and ccomp, iobj	61.5
advcl , appos, csubj, abbrev, infmod, npavmod, rel, acomp, agent, csubjpass, partmod, pobj, purpcl, xsubj	63.5

Table 5.1: Ablation Test for Significant Relations

In *Table 5.1,* we find that leaving out *Dep* and *Rcmod* causes significant accuracy improvement, rather than including all the relations. But, we still cannot be sure which among *Dep* and *Rcmod* plays the spoilsport. So we perform another experiment in a different domain involving only these two relations.

In *Table 5.2*, we find that Dep causes the real problem. This is also intuitive when we see the definition of the *Dep* relation in Stanford Dependencies Manual which says "*dependency is labeled as dep when the system is unable to determine a more precise*

dependency relation between two words". Thus it captures many stray relations and introduces noise in the graph.

Relation Set	Accuracy
With Dep+Rcmod	66
Without Dep	**69**
Without Rcmod	67
Without Dep+Rcmod	68

Table 5.2: Ablation Test for Dep and Rcmod

b. Merging Threshold

Any 2 feature clusters are merged if the inter-cluster distance is less than some threshold distance θ. The distance is measured as the number of edges in the shortest path connecting the 2 cluster-heads. If θ is very small, then any 2 clusters having some long-range dependency will not be merged. Whereas if θ is very large, then all the features will be merged and feature specific dependencies will be lost. We used a small seed set from an arbitrary domain to find the optimal value of θ and cross-validated it across other domains.

θ	Accuracy (%)
2	67.85
3	**69.28**
4	68.21
5	67.40

Table 5.3: Inter-Cluster Distance Threshold Accuracy

Table 5.3 indicates that $\theta = 3$ will give the optimal result. $\theta = 2$ means all the clusters are disjoint and there is no merging, whereas $\theta = 3$ implies any 2 clusters are merged if there is only one intermediate word linking them.

5.5 EXPERIMENTAL EVALUATION

We used 3 datasets. $Dataset_1$ consisted of 500 reviews extracted from the dataset used by Lakkaraju *et al.* (2011). The extracted data came from 3 domains *laptops, camera and printers*.

The second dataset was extracted from the data used by Hu and Liu *et al.* (2004). It consisted of about 2500 reviews from varied domains like *antivirus, camera, dvd, ipod, music player, router, mobile etc*. Each sentence is tagged with a feature and sentiment orientation of the sentence with respect to the feature.

In the original dataset (*Hu and Liu, 2004*), majority of the sentences consisted of a single feature, and had either entirely positive or entirely negative orientation. From there a new dataset was constructed, by combining each positive sentiment sentence with a negative sentiment sentence using connectives, *in the same domain, having the same set of features.* This forms our $Dataset_2$. Now, each sentence in this new dataset has a mixed emotion about various features.

We determined $Baseline_1$ by counting the number of positive and negative opinion words in the sentence. The final polarity is determined by majority voting. This is a very naïve baseline. So we define a $Baseline_2$ (*Hu and Liu et al., 2004*). If there 'n' features f_i and 'm' opinion words O_i, each O_i expresses an opinion about the nearest feature f_i.

Domain	Baseline 1 (%)	Baseline 2 (%)	Proposed System (%)
Antivirus	50.00	56.82	**63.63**
Camera 1	50.00	61.67	**78.33**
Camera 2	50.00	61.76	**70.58**
Camera 3	51.67	53.33	**60.00**
Camera 4	52.38	57.14	**78.57**
Diaper	50.00	63.63	**57.57**
DVD	52.21	63.23	**66.18**
IPOD	50.00	57.69	**67.30**
Mobile 1	51.16	61.63	**66.28**
Mobile 2	50.81	65.32	**70.96**
Music Player 1	50.30	57.62	**64.37**
Music Player 2	50.00	60.60	**67.02**
Router 1	50.00	58.33	**61.67**
Router 2	50.00	59.72	**70.83**

Table 5.4: Domain Specific Accuracy for Our Rule Based System in $Dataset_2$

We used the sentiment lexicon used by Hu and Liu *et al.* (2004) for rule based classification. Since we have a 2-class classification (positive or negative), any tie is resolved by flipping a coin.

Table 5.4 gives the domain specific accuracy comparison of our system with $Baseline_1$ and $Baseline_2$. We find that the proposed system performs way better than both the baselines in *every domain*. *Table 5.5* gives the average accuracy of the system and the baselines across all the domains.

System	Accuracy (%)
$Baseline_1$	50.35
$Baseline_2$	58.93
Proposed System	**70.00**

Table 5.5: Overall Accuracy for Our Rule-Based System in $Dataset_2$

We also performed comparisons with another state-of-the-art system namely, CFACTS developed by Lakkaraju *et al.* (2011). The CFACTS system extracted features as well as performed the feature specific analysis whereas our system performed only the feature specific analysis given the product specific feature. So we compared only the sentiment evaluation of the 2 systems. This is a valid comparison as CFACTS claimed to have a 100% topic purity in feature extraction which means its feature extraction accuracy cannot influence its sentiment evaluation accuracy.

The performance comparison between the feature specific module of CFACTS and our system is made under the *assumption that the features should be explicitly present in the review*. This is necessary in our system as the user is providing the feature with respect to which the review has to be analyzed.

Consider the review sentence, "*The mobile is too heavy*". Here the implicit feature is *weight* and the implicit sentiment is *negative*. Since the system, we developed, does not use any domain specific data for sentiment classification, such reviews cannot be aptly handled by the system. The seed set of data used for learning the parameters was *domain independent* and the learning was a one-time affair. Hence, these kinds of reviews were ignored while performing comparisons.

From *Table 5.6*, we find that the proposed system performs at par with all the unsupervised systems, mentioned above, *with no data requirement*. This, however, comes at a cost that it cannot capture domain-specific implicit feature or hidden sentiment.

In order to have a flavor of the system performance, when tagged data is available, we performed experimental evaluations in 2 arbitrary domains namely, *camera and mobile* using $Dataset_1$.

System	Sentiment Evaluation Accuracy (%)
$Baseline_1$	68.75
$Baseline_2$	61.10
CFACTS-R	80.54
CFACTS	**81.28**
FACTS-R	72.25
FACTS	75.72
JST	76.18
Proposed System	**80.98**

Table 5.6: Sentiment Classification Accuracy Comparison for Rule-Based Classification in $Dataset_1$

Domain	Baseline 1 (%) Accuracy (Precision/Recall)	Proposed System (%) Accuracy (Precision/Recall)
Mobile	51.42 (50.72/99.29)	**83.82 (83.82/83.82)**
Camera	50	**86.99 (84.73/90.24)**

Table 5.7: Supervised Classification Accuracy in Two Domains in $Dataset_2$

The supervised system uses Support Vector Machines for classification of feature vectors. *Table 5.7* shows the huge leap in accuracy from the naïve baseline. The difference in accuracy between the rule-based system and the supervised classification system stems from the fact, that the system can now capture both domain specific sentiment and implicit features. But this comes at a cost of enhanced data tagged data requirement for every domain and the system needs to be trained separately for every domain.

5.6 SUMMARY

We showed that incorporating feature specificity *significantly improves accuracy over the baseline*. The sentiment orientation in product reviews is seldom entirely positive or entirely negative. It consists of multiple features with different sentiment orientations. The proposed

system clusters the feature specific information and finds it polarity. We performed extensive evaluations across various domains in product reviews to validate our claim. Accuracy improvement was significant in *all the domains*. We also showed that marked improvement is possible in case of supervised classification systems using tagged data. The system has minimal data requirement since it requires a *one-time learning* of its parameters, which are *domain independent*. We compared our system with 2 state-of-the-art systems and showed that it achieves significant accuracy improvement over one and performs at par with the other despite its data limitations.

The drawback of the system is that it cannot evaluate domain dependent implicit sentiment as it does not train on any domain specific data. Thus the system does not distinguish between "*The story is unpredictable*" (positive sentiment) and "*The steering wheel is unpredictable*" (negative sentiment). This is due to the usage of a generic sentiment lexicon, in the final stage, in lexicon based classification. Supervised classification can distinguish between the 2 sentiments but it needs tagged data and separate training for every domain.

Chapter 6

6. TWISENT: A MULTI-STAGE SYSTEM FOR ANALYZING SENTIMENT IN TWITTER

This chapter analyzes the effect of informal language form and the social media content for micro-blog sentiment analysis. We suggest a multi-stage system for sentiment analysis in *Twitter*, which consists of specialized modules to handle the nuances of micro-blogging genres. The roadmap to the rest of this chapter is as follows: *Section 6.1* gives the system architecture. *Section 6.2* presents the API for *tweet* extraction and lexicon-based polarity detection in the final *tweet extract*, output by the final module .of *TwiSent*. *Section 6.3* categorizes spam in the social media, in context of sentiment analysis and presents an *expectation-maximization* approach to detect the same under different settings. *Section 6.5* discusses different spelling errors and abbreviations encountered in the social media and presents a heuristically-driven minimum edit-distance based spell-checker to handle the same. *Section 6.6* presents an entity specific algorithm to extract sentiment from tweets with respect to a given entity. *Section 6.7* presents experimental evaluations in two datasets followed by discussion in *Section 6.8*.

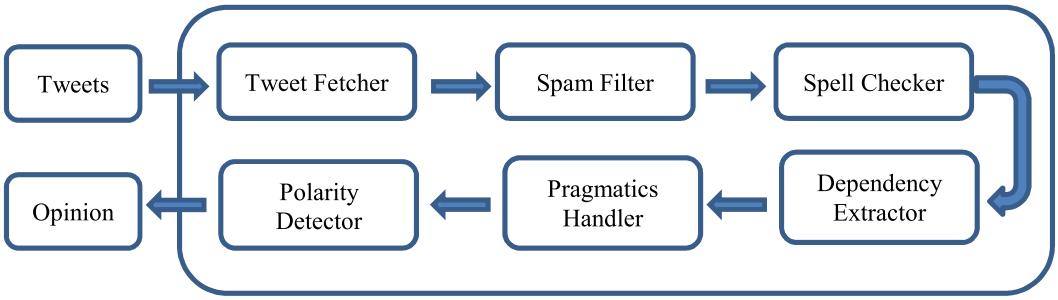


Figure 6.1: TwiSent Architecture Diagram

6.1 TWITTER CHARACTERISTICS

Twitter is an extremely noisy medium owing to the inherent text limitation of 140 characters per tweet. People use all sorts of abbreviations, slangs, acronyms to convey their opinions which are difficult to process by standard natural lanuguage processing tools. The informal language prevalent in the Twitter and spams pose a real problem to Sentiment Analysis of the

tweets. In this chapter, we categorize the different types of spams encountered in Twitter with respect to Sentiment Analysis like *bot-generated tweets, fake url's, abundance of hashtags, promotional tweets etc.* Thereafter, we propose an algorithm based on Twitter-specific features like *hashtags, @users, slangs, acronyms, abbreviations etc.* encountered in Twitter to detect spams. We categorize the different types of abbreviations and common spelling errors encountered in the social media like *gr8, btfl, awsum, teror etc.* The intensity of the user sentiment in the *tweet* is often strengthened by the repetation of characters and capitalization like *happyyyyyyy, HATE etc,* which form a part of *Pragmatics*. We propose a simple rule-based module to preserve the Pragmatics in *tweets*.

6.2 SYSTEM ARCHITECTURE

In this section we give an overview of the complete system and define the functionality ofeach module. *Figure 6.1* presents the architecture of the system. **Tweet Fetcher** fetches tweets based on the user-given search string. **Spam Filter** filters the spams providing clean tweets to the next module. **Spell-Checker** performs text cleaning by correcting the spelling and does text normalization. **Extract Dependency** extracts opinion words from the tweet referring only to the search string. **Pragmatics Handler** handles pragmatics to capture the intensity of emotions. **Polarity Predictor** predicts the overall polarity of the tweet (*positive, negative or objective*) based on the input evidence available from the previous modules. It also gives a classification score based on the sentiment of all the tweets referenced by the topic.

6.3 TWEET FETCHER AND POLARITY DETECTOR

The Tweet Fetcher extracts tweets based on the search string entered by the user. A Twitter API[9] is used to obtain live feeds from Twitter. Based on the search string, we retrieve the latest 200 tweets in English. Many of these tweets are either spams or irrelevant to the search string and are subsequently removed. In the scenario where the number of clean tweets falls below 100, the process is repeated. The tweets are in XML format which needs to be parsed to extract the tweet bodies.

We follow two different approaches for polarity detection. The polarity detector is modeled as: a *lexicon-based* classifier and a *supervised* classification system. In the lexicon-based classification, a lexicon is used to classify the final set of words output by the

[9] http://search.Twitter.com/search.atom

Pragmatics Handler into *positive, negative or the objective class*. The Bing Liu sentiment lexicon[10] is used to classify the final set of opinion words output by the Pragmatics module. It contains around 6800 words which are manually polarity labeled. The final polarity of the tweet is given by the majority voting of the polarity of the opinion words in the *tweet extract,* output by the final stage of *TwiSent*. In the supervised system, a classifier[11] is used to classify the tweet extract using bag-of-words features. Unigram and bigrams in the tweet extract are used as features in SVM's, ignoring the StopWords.

6.4 SPAM FILTER

Spam is the use of electronic messaging systems to send unsolicited bulk messages indiscriminately[12]. Another definition[13] says that Spam is flooding the Internet with many copies of the same message, to force the message on people who would not otherwise choose to receive it. Jindal *et al.* (2008) identify three types of spams in sentiment analysis: Untruthful opinions (undeserving positive reviews), reviews on brands only (and not on the specific product) and non-reviews (advertisements and other irrelevant reviews containing no opinions). However, we provide, here, a more detailed categorization of Twitter spams:

1. *Re-tweets* - A re-tweet is the same tweet posted by some other person. We consider these as spam as they do not contribute anything new to the data.

2. *Promotional tweets for some entity* - These tweets are usually sponsored by the companies to popularize their products.

3. *Tweets containing links to some other websites* - In many cases these are promotional and do not contain genuine human sentiment.

4. *Tweets in languages other than English* - We assume that foreign languages are incomprehensible by the users and machine.

5. *Tweets with incomplete text* - A tweet length limit of 140-characters is, many a times, not enough to express some concepts completely. This leads users to break the sentence into multiple tweets. If seen individually, these tweets would not make complete sense.

6. *Automatically generated tweets by bots* - These, too, are generally promotional tweets and are the most common form of spamming.

[10] http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar
[11] http://svmlight.joachims.org/
[12] http://en.wikipedia.org/Spam_(electronic)
[13] http://spam.abuse.net/overview/whatisspam.shtml

7. *Tweets built primarily for search engines or tweets with excessive off-topic keywords* - They aim to have maximum chances of a hit.

8. *Multiple tweets offering substantially the same content* - They are generally created using a fixed template, in which only a keyword is changed to generate multiple similar tweets.

The list is not exhaustive as new categories of spams are generated regularly. Therefore, the best way to define spam in the field of sentiment analysis pertaining to Twitter is to say that '*a Twitter spam is any tweet that does not contribute any genuine sentiment i.e. it should not belong to any positive, negative or objective category with respect to the entity searched*.' The underlying logic behind this assumption is that the definition of a sentiment-bearing tweet is invariant with time, while that of a spam changes.

6.4.1 A Partially Supervised Approach to Spam Filter using Expectation Maximization

Most machine learning methods treat the learner as a passive recipient of the data to be processed (Cohn *et al.*, 1996). The problem with any supervised approach is that the algorithm might be based on *specific* examples that are present in the training corpus. In Twitter newer variants of spams are encountered frequently. Thus, adaptation of the algorithm to these new instances of spams requires human supervision. We adopt a partially supervised approach to alleviate this problem. In this setting, we have labeled training examples of only one category, namely the *non-spam* class, and a mixed set of unlabeled examples containing *spams* as well as *non-spams*. A classifier is trained on this set which tries to identify the non-spam tweets out of the mixed bag. The approach discussed here uses Naive Bayesian text classification to implement a partially supervised learning based on *Expectation Maximization* (Liu *et al.*, 2002).

Input: Build an initial naive bayes classifier NB- C, using the tweet sets *M* (mixed unlabeled set containing spams and non-spams) and *P* (labeled non-spam set)

1: Loop while classifier parameters change

2: for each tweet $t_i \in M$ do

3: Compute $Pr[c_1 |t_i]$, $Pr[c_2 |t_i]$ using the current NB //c_1 - non-spam class , c_2 - spam class

4: $Pr[c_2 |t_i]= 1 - Pr[c_1 |t_i]$

5: Update $Pr[f_{i,k}|c_1]$ and $Pr[c_1]$ given the probabilistically assigned class for all t_i ($Pr[c_1|t_i]$). (a new NB-C is being built in the process)

6: end for

7: end loop

Algorithm 6.1: Spam Filter Algorithm

Given that a tweet t_i can be represented by a feature set F - $\{f_1, f_2, \ldots f_n\}$ and a predefined set of classes $\{c_1, c_2, \ldots, c_m\}$, by Naive Bayes theorem, we have, $\Pr[c_j] = \sum_i \Pr[c_j|t_i] / |T|$ where, t_i is the i^{th} tweet in the corpus, and $|T|$ is the total number of tweets present in the corpus. By Naive Bayes conditional independence assumption,

$$\Pr[c_j | t_i] = \frac{\Pr[c_j] \prod_k \Pr[f_{i,k}|c_j]}{\sum_r \Pr[c_r] \prod_k P(f_{i,k}|c_r)}$$

where, $f_{i,k}$ is the k^{th} feature of the i^{th} tweet. The class with the highest $P(c_j \mid t_i)$ is assigned as the class label of the tweet.

The algorithm begins with assigning all the samples in the non-spam class P as non-spam, and all the samples in the mixed unlabeled set M as spam. In the first iteration, all the feature values are calculated. The class probabilities are calculated considering individual feature values leading to the tweet probability for each class.

We follow two different approaches while implementing this algorithm. In the first approach, the *Positive, Negative and Objective* training samples are collectively marked as non-spam. Thus we have information about the non-spam class collectively which makes it a two-class classification problem (considering spam as another class). In the second approach, *Positive, Negative, Objective-not-Spam* and *Objective-Spam* samples are considered as separate classes, which make it a four-class classification problem. In a *two-class setting* all the tweets in the mixed set M, which are more probable to be in the non-spam class than in the spam class, are reassigned to the set P. In a *four-class setting*, a tweet is reassigned from the spam category to one of the three classes (positive, negative and objective) for which the probability is highest, if the difference between the probability for this class and the spam class is greater than a threshold. The threshold was decided empirically. The algorithm halts when there is no further reassignment to any other category. The algorithm, then, moves into the next iteration. *Algorithm 1* gives the algorithm pseudo-code.

6.4.2 Spam Filter Features

The following set of features is used in the spam filter module:

1. *Number of Words per Tweet*	**9.** *Frequency of Foreign Words*

2. *Average Word Length* **3.** *Frequency of "?" and "!"* **4.** *Frequency of Numeral Characters* **5.** *Frequency of hashtags* **6.** *Frequency of @users* **7.** *Extent of Capitalization* **8.** *Frequency of the First POS Tag*	**10.** *Validity of First Word* **11.** *Presence / Absence of links* **12.** *Frequency of POS Tags* **13.** *Strength of Character Elongation* **14.** *Frequency of Slang Words* **15.** *Average Positive and Negative Sentiment of Tweets*

Table 6.1: Features Used in the Spam Filter

Two-class Setting

After the feature values have been initiated, the algorithm runs till there is a change in the total number of non-spam tweets. Each iteration loops over all the tweets. It begins with the calculation of various feature probabilities for individual tweets. Once all the feature probabilities have been calculated for a tweet, the class probability for that tweet is calculated. This is done for all the tweets. Once this loop is over, the tweets in the spam class, having probability of being in the non-spam class more than a *threshold* value, are reassigned to the non-spam category. The iteration ends with recalculation of the class parameters for each feature. At the end of the algorithm, we have the final classification into spam and non-spam classes.

Four-class Setting

Unlike the two-class case, in this setting, there are four classes to consider. The only difference in the training phase, here, is that instead of calculating the class parameters for features of non-spam class, now we need to calculate it separately for all the three classes (*positive, negative and objective*). The probability calculations for individual features and the tweet as a whole follow the same approach as discussed above. The difference lies in the reassignment of the tweets in the mixed set, and the halting criteria. A tweet is reassigned from the spam category to one of the three classes, *positive, negative, and objective*, for which the probability is highest, if the difference between the probability for this class and the spam class is greater than a threshold. The algorithm halts when the number of spam tweets becomes constant.

The algorithm begins with assigning all the samples in the non-spam class P as non-spam, and all the samples in the mixed unlabeled set M as spam. In the first iteration, all the feature values are calculated. The class probabilities are calculated considering individual feature values leading to the tweet probability for each class.

Algorithm 6.1 gives the algorithm pseudo-code.

6.4.3 Spam Filter Features

The following set of features is used in the spam filter module:

Number of Words per Tweet

The number of words is included as a feature to remove those tweets that are too small or too large as spams tend to use the entire length limit of 140 characters. The *average number of words per Tweet* is calculated for the *non-spam* class during each iteration. The *closer* the number of words in a Tweet is to this average, the *more probable* it is to be in the *non-spam* category. The probability of this feature given the tweet for *non-spam* class is calculated as:

$$\Pr[numwords \mid tweet_i] = 1 - \frac{numwords_i - avgNumWordsperTweet}{maxWordsperTweet - avgNumWordsperTweet}$$

Average Word Length

Spammers might use too long words in order to avoid being caught based on the standard set of words. The probability value of the feature is calculated as:

$$\Pr[avgWordLength \mid tweet_i] = 1 - \frac{avgWordLength_i - avgWordLength}{maxAvgWordLength - avgWordLength}$$

Frequency of "?" and "!"

These are two separate features which have been included to take into account the differences between a *true expression* and a *spam*. The probability values are calculated in a similar fashion as mentioned in the above cases. The expression looks like:

$$\Pr[Ques \mid tweet_i] = 1 - \frac{numQues_i - avgNumQues}{maxNumQues - avgNumQues}$$

A similar expression evaluates the probability for exclamation marks too.

Frequency of Numeral Characters

Based on observations, it is found that promotional tweets tend to use numeral characters more frequently than normal tweets. This forms the basic ideology behind this feature.

Frequency of hashtags

Hashtags are a unique concept in Twitter, where in, users use hashtags to highlight the entity that the tweet is talking about. Using one or two hashtags in a tweet is common, but some spam tweets, particularly the ones that are meant for high ranking in search, contain numerous hashtags in the same tweet.

Frequency of @users

@users are another innovation in Twitter, using which, users can tag other users in a tweet to signify that they are talking about them. Spammers utilize this feature by tagging multiple users in a tweet and adding some promotional text or some link which they, then, tend to follow.

Extent of Capitalization

Users tend to use upper-case characters in order to emphasize on something. Spammers misuse it by many times capitalizing the complete tweet so as to catch the eye of the readers.

Strength of Character Elongation

This is one feature that we generally do not see in spams. Users use elongated forms of words, e.g. *haaaaaaaappppyyyyyy* in place of *happy* to stress on that particular emotion.

Frequency of POS Tags

All the tweets are tagged with the Stanford POS tagger. The Stanford POS Tagger follows the Penn TreeBank tag set which has 36 tags defined. A 37th tag field is added to account for punctuations and unknown words. The average individual tag frequencies are calculated to aid the probability calculations. Note that these are a set of 37 separate features and not just a single feature.

Frequency of the First POS Tag

It is observed that promotional tweets have a higher tendency to start with verbs, as compared to other POS tags. This made us include this as a separate feature.

Frequency of Foreign Words

One of the common types of spam is tweets in other languages. To account for these, we added the frequency of foreign words as a feature. Any word that is not present in WordNet 2.1 and the dictionaries[14] [15] is considered as a word of foreign language.

[14] http://reference.dictionary.com/

Frequency of Slang Words

A typical Twitter user has a much more tendency to use slangs than a spammer.

Validity of First Word

It is observed that many spam tweets have an incomplete first word. A crude way to check whether the first word is complete is to check if it is in English. Basically, if the first word is not in English, then we assume it to be incomplete.

Average Positive and Negative Sentiment of Tweets

The underlying assumption for this feature is that tweets of the same category exhibit similar sentiments. The sentiment scores for each word in a tweet are calculated using SentiWordNet. These scores are then averaged over the word count of the tweet to get the average sentiment score of the tweet.

Presence/ Absence of links

A spam tweet has a higher tendency to contain a link than a non-spam tweet. This feature is a Boolean which is true if a link is present in the tweet and false otherwise.

The probability expression for all the features is computed similar to that of the features 1, 2 and 3. In each iteration of the algorithm, these values are recalculated for each tweet. The product of all these probabilities gives us the probability of the tweet being in the spam/non-spam class. Based on this probability value, the tweet can be classified to the respective class.

6.5 SPELL CHECKER AND TEXT NORMALIZATION

Most spell-checkers are not effective in handling noisy text in informal language. They fail miserably for the *conversational* language used in social media or internet chats. For example, the spelling suggestions for *awesum* by Open-Office[16] and Microsoft Word 2003[17], ranged over *awe sum, awe-sum, awesum, awes-um, awesome, awestruck, awestricken, aweigh* and that for *gr8* as *gr, gr 8*. Thus we attempt to incorporate a Spell Checker in TwiSent that addresses the common types of abbreviations, linguistic phenomena commonly observed in the social media. This is done to have maximum coverage during lexicon matching.

[15] http://www.urbandictionary.com/
[16] http://www.openoffice.org/
[17] http://office.microsoft.com/en-us/

6.5.1 Types of Common Spelling Errors in Social Media

This section gives an overview of some of the most prevalent abbreviations, spelling variations and textual errors in any Social Networking Site like the Twitter. The list is compiled from the tagged tweets for this work and from Bieswanger (2007).

1. Dropping of Vowels - Example: *btfl (beautiful), lvng (loving)*
2. Vowel Exchange - Exchange between pairwise vowels due to phonetic similarity. Example: *good vs. gud (o,u)*
3. Mis-spelt words - Example: *redicule (ridicule), magnificant (magnificent)*
4. Text Compression - Example: *shok (shock), terorism (terrorism)*
5. Phonetic Transformation - Example: *be8r (better), gud (good), fy9 (fine), gr8 (great)*
6. Normalization and Pragmatics - Example: *hapyyyyyy (happy), guuuuud (good)*
7. Segmentation with Punctuation - Example: *beautiful, (beautiful)*
8. Segmentation with Compound Words - Example: *breathtaking (breath-taking), eyecatching (eye-catching), good-looking (good looking)*
9. Hashtags and Segmentation - Example: *#notevenkidding, #worthawatch*
10. Combination of all - Example: *#awsummm (awesome), gr88888 (great), amzng,btfl (amazing, beautiful).*

6.5.2 Spell Checker Algorithm

In this section, we give a heuristically driven spell checker algorithm to resolve the *identified errors* with a *minimum edit distance based spell checker*. A lexicon of all opinion words is created which comprises of all the words from the sentiment specific resource SentiWordNet (Esuli *et al.*, 2006).

The *normalize* function in *Algorithm 6.2* takes care of *Pragmatics* (*Section 6.5*) and *Number Homophones.* It replaces every letter occurring two times consecutively by a single letter. Example: *happpyyyy* with *hapy*. It also replaces number homophones with their alphabetic variants. Example: *'2' with 'to', '8' with 'eat', '9' with 'ine'*.

The *vowel_dropped* function returns the string with all the vowels removed. This is to take care of the *vowel dropping* phenomenon. The function *overlapping_characters* computes the number of characters, in order, that matches between a word in the set of suggestions and the current string. Higher the number of overlapping characters of a suggestion with a given word, higher is its chance of winning.

The parameters *offset* and *adv* are determined empirically. The words are marked during normalization, so that their pragmatics is aptly handled in the next module. This is to ensure that *happppyyyyy,* normalized to *hapy* and thereafter spell-corrected to *happy,* is marked so as to not lose its pragmatic content from multiple occurrences of a letter.

Algorithm 2: An Example

Consider the word *s=guuuuud*. The suggestion list *S* will consist of all letters starting with *g*. Consider *w=good* $\in S$. Here, *w'=gd*, *s'=gud*, diff(*gud,gd*)=1, offset=7.

score[*good*] = min(edit_distance(*guuuuud,good*),

edit_distance(*guuuuud,gd*),edit_distance(*gud,good*))

min(3, 5, 1) = 1. *good* with score=1 will be among the top *m* entries and retained to be processed in the next stage for phonetic transformation. min_edit=edit$_5$ =edit_distance(*guud, gud*)=1, count = 2, final_score = 1 + 0 + 1 − 2 = 0 (*exact match*).

6.6 HANDLING PRAGMATICS

Pragmatics[18] is a subfield of linguistics which studies how the transmission of meaning depends not only on the linguistic knowledge (e.g. grammar, lexicon *etc.*) of the speaker and listener, but also on the context of the utterance, knowledge about the status of those involved, the inferred intent of the speaker *etc.*

Input: For string s, let *S* be the set of words in the lexicon starting with the initial letter of *s*.

```
/* Module Spell Checker */
for each word w ∈ S do
  w'=vowel_dropped(w)
  s'=normalize(s)
/*diff(s,w) gives difference of length between s and w*/
  if diff(s' , w') < offset then
   score[w]=min(edit_distance(s,w),edit_distance(s,
   w'), edit_distance(s' , w))
  else
    score[w]=max_centinel
  end if
end for
```

Sort score of each *w* in the Lexicon and retain the top m entries in suggestions(*s*) for the original string *s*

[18] http://en.wikipedia.org/wiki/Pragmatics

```
for each t in suggestions(s) do
  edit1=edit_distance(s' , s)
/*t.replace(char1,char2) replaces all occurrences of char1 in the string t with char2*/
  edit2=edit_distance(t.replace( a , e), s')
  edit3=edit_distance(t.replace(e , a), s')
  edit4=edit_distance(t.replace(o , u), s')
  edit5=edit_distance(t.replace(u , o), s')
  edit6=edit_distance(t.replace(i , e), s')
  edit7=edit_distance(t.replace(e, i), s')
  count=overlapping_characters(t, s')
  min_edit=
  min(edit1,edit2,edit3,edit4,edit5,edit6,edit7)
  if (min_edit ==0 or score[s'] == 0) then
    adv=-2 /* for exact match assign advantage score */
  else
    adv=0
  end if
  final_score[t]=min_edit+adv+score[w]-count;
end for
return t with minimum final_score;
```

Algorithm 6.2: Spell Checker Algorithm

Few works have addressed the issue of pragmatics in natural language processing applications and fewer in the area of social media. In this work, we identify some of the pragmatics embedded in the tweets and suggest an approach to handle them, though naively. We identified the different forms of pragmatics in Twitter and actions on them as:

1. *Happiness, joy or excitement is often expressed by elongating a word, repeating alphabets multiple times* - Example: *happppyyyyyy*, *goooooood*.
These forms are given more weightage than other words by repeating them twice.

2. *Use of Hashtags* - Example: *#overrated, #worthawatch*. These forms are treated by Segmentation to retrieve valid tokens which are given more weightage than other commonly occurring words by repeating them thrice.

3. *Use of Emoticons* is common in social media and micro-blogging sites where the users express their sentiment in the form of accepted symbols. Example: ☺ *(happy)*, ☹ *(sad)*

4. Happiness, joy, sorrow, hatred, enthusiasm, excitement, bewilderment etc. are also commonly expressed by Capitalization where words are written in capital letters to express intensity of user sentiments.

Full Caps - Example: *I HATED that movie.* These forms are given more weightage than other words by repeating them thrice.

Partial Caps- Example: *She is a Loving mom*. These forms are given more weightage than other words by repeating them twice.

6.7 ENTITY SPECIFICITY

A *tweet* may have multiple entities with a different opinion expression about each entity. Thus it is important to extract the opinion expression relating to a particular entity. The tweet, "*The film bombed at the box office although the actors put up a reasonable performance",* is negative w.r.t *film* but positive w.r.t *actors.*

Mukherjee *et al.* (2012) propose a method in which *Dependency Parsing* (Marie-Catherine *et al.*, 2006, 2008) is used to capture the association between any specific feature and the expressions of opinion that come together to describe that feature. The underlying hypothesis is that: *More closely related words come together to express an opinion about a feature.*

The association between features and opinion expressions are captured by the *short range* and *long range dependencies* between the words using dependency parsing. They achieved a high accuracy across all domains over the baseline and comparable accuracy to state-of-the-art systems.

The reason for implementing this specific algorithm is two-fold: 1) Tweets often have a variable word order and syntactic discontinuities. Dependency based parsing allows adequate treatment of languages with variable word order, where discontinuous syntactic constructions are more common (Mel'ˇcuk 1988; Covington 1990). 2) The algorithm has minimal data requirements. It requires a one-time training of its *domain-independent* parameters. In this work, we directly incorporated the parameter set used in their work.

6.7.1 Entity Specific Algorithm

Consider a sentence *S* and 2 consecutive words $w_i, w_{i+1} \in S$. If $w_i, w_{i+1} \notin StopWords\_List$, then they are directly related. This helps to capture *short range dependencies*. Let *Dependency_Relation* be the list of significant dependency parsing relations (like *nsubj, dobj,*

advmod, amod etc.). Any 2 words w_i and w_j in S are directly related, *if,* $\exists D_l \; s.t \; D_l(w_i, w_j) \in Dependency\_Relation$. This helps to capture *long range dependencies*. The direct neighbor and dependency relations are combined to form the master *relation set* ***R***.

Given a sentence *S*, let ***W*** be the set of all words in the sentence. A Graph $G(W, E)$ is constructed such that any $w_i, w_j \in W$ are directly connected by $e_k \in E$, *if* $\exists R_l \; s.t \; R_l(w_i, w_j) \in R$. All the Nouns in the given tweet are extracted by a POS-Tagger which form the feature set *F*. Let $f_t \in F$ be the target feature *i.e.* the feature with respect to which we want to evaluate the sentiment of the sentence.

Let there be *'n'* features where *n* is the dimension of *F*. We initialize '*n'* clusters C_i, corresponding to each feature $f_i \in F$ *s.t.* f_i is the clusterhead of C_i. We assign each word $w_i \in S$ to the cluster whose clusterhead is closest to it. The distance is measured in terms of the number of edges in the shortest path, connecting any word and a clusterhead. Any 2 clusters are merged if the distance between their clusterheads is less than some threshold. Finally, the set of words in the cluster C_t, corresponding to the target feature f_t gives the opinion about f_t (*refer to Algorithm 5.1*)**.**

Consider the review "*Nokia's battery life is quite good but that of Micromax sucks".* ***F***={ *Nokia, battery, life, Micromax* }forms the initial feature set. The target feature is $\boldsymbol{f_t}$ =*Nokia*. The graph consists of all the words in the sentence as vertices. All the words are connected by relations defined by the master relation set ***R*,** consisting of dependency relations and direct neighbor relations. The target cluster $\boldsymbol{C_t}$ has the clusterhead $\boldsymbol{f_t}$. {*quite, good*} are closest to *life* and are assigned to its cluster. Now, {*Nokia, life*} are related by the dependency relation *possession modifier* whereas {*battery, life*} are related by the relation *noun compound modifier*. The intercluster distance between {*battery, Nokia*} is one which is less than the threshold=3 and thus the clusters are merged. Finally $\boldsymbol{C_t}$ comprises of {*battery, life, quite, good*} which represents the opinion expression about the target feature *Nokia*.

6.8 EXPERIMENTAL SETUP

Two datasets are used for evaluation. The first dataset comprises of 8507 tweets manually annotated whereas the second dataset comprises of 15,214 tweets automatically annotated using hashtags. Twitter was crawled using Tweet Fetcher and 8507 tweets were collected based on a total of around 2000 different entities from 20 different domains (refer to *Table 6.2*). These were manually annotated by 4 annotators into four classes *positive, negative, objective-not-spam and objective-spam*. The objective-not-spam category contains tweets

which are objective in nature but are not spams. The objective-spam category contains tweets which are spams, following our categorization of spams. This forms our gold standard data.

Movie, Restaurant, Television, Politics, Sports, Education, Philosophy, Travel, Books, Technology, Banking & Finance, Business, Music, Environment, Computers, Automobiles, Cosmetics brands, Amusement parks, Eatables, History

Table 6.2: Dataset Domains

The Twitter API was used to collect another set of 15,214 tweets based on *hashtags*. Hashtags *#positive, #joy, #excited, #happy etc* were used to collect tweets bearing positive sentiment, whereas hashtags like *#negative, #sad, #depressed, #gloomy, #disappointed etc*. were used to collect negative sentiment tweets. *Subsequently, these hashtag keywords were removed from the tweets.*

Manually Annotated Dataset				
#Positive	#Negative	#Objective Not Spam	#Objective Spam	Total
2548	1209	2757	1993	8507
Auto Annotated Dataset				
#Positive		#Negative		Total
7348		7866		15214

Table 6.3: Dataset Statistics

6.8.1 Data Pre-Processing

The crawled tweets were pre-processed in the following way:

Link and User Normalization

All the links (urls) in the tweets were replaced by #link.

All the user id's in the tweets were replaced by #user.

Chat Lingo Normalization

There are some standard abbreviations and lingo used in the tweets. A dictionary[19] was used to map these words to their proper words in the lexical resources. This step actually aids the spell-checker in the subsequent module.

[19] http://chat.reichards.net/

Emoticon Handler

Emoticons and smileys are very commonly used in the tweets to express subtle emotions. In most cases, they directly give away the sentiment of the user. An emoticon dictionary[20] was used to map each emoticon to *positive* or *negative* class. Thereafter, each emotion was replaced with *#emoticon_class*. Any tweet containing these smileys is directly classified into the respective class of the emoticon. This heuristics works well in most cases, except the *sarcastic* tweets which are very difficult to detect.

Stop Word Remover

A stopwords list[21] was used to remove all the stopwords in the tweet like *but, though, it, and, is etc*.

Negation Handling

The following negation operators like *no, never, not, neither, nor* were used and all the words in a context window of look-ahead 5 from the occurrence of any of these operators were marked. In the polarity detection phase, the polarity of all these marked words was reversed.

Stemming

Lovins stemmer (Lovins, 1968) is used to stem the words so that *happy, happiest, happier* are reduced to the root word *happy*.

6.8.2 Resources

Stanford Dependency Parser[22] is used for extracting dependency relations in the Entity Specific Module. The Spam Filter uses Standford POS Tagger (Tou, 2000) and WordNet 2.1 (Fellbaum 1998) for finding the validity of words and POS tags.

The Bing Liu sentiment lexicon[23] containing around 6800 words, which are manually polarity labeled, is used to classify the final set of opinion words output by the Pragmatics module. The final polarity of the tweet is given by the majority voting of the polarity of the opinion words in the *tweet extract* output by the final stage of *TwiSent*.

[20] http://chat.reichards.net/smiley.shtml
[21] http://www.ranks.nl/resources/stopwords.html
[22] nlp.stanford.edu:8080/parser/index.jsp
[23] http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar

6.8.3 Baseline System

We compare our system performance on both the datasets to *C-Feel-It* (Joshi *et al.*, 2011). It is a rule-based system which classifies a tweet as positive or negative based on the opinion words present in it. It uses a weighted polarity scoring with four sentiment lexicons. C-Feel-It employs a *similar data pre-processing strategy*, but does *not* consist of specific modules for *spam filtering, text normalization, entity specificity and pragmatics handling*. Our work extends their idea of twitter sentiment analysis and gives it a new dimension.

6.8.4 Spam Filter Evaluation

Evaluation of the Spam Filter module is first done, independent of the remaining system, under a two-class and a four-class setting (refer to *Table 6.4* and *Table 6.5*). This evaluation is done on *Dataset 1*. Three cases are considered: one with a complete corpus, another one with only spams and one with only non-spams. Evaluation of the two settings is done based on the precision and recall values. Precision and recall values are defined as:

$$precision = \frac{Number\ of\ correctly\ classfied\ tweets}{Total\ number\ of\ tweets\ in\ the\ corpus}$$

$$recall = \frac{Number\ of\ correctly\ classified\ \boldsymbol{spam}\ tweets}{Total\ number\ of\ \boldsymbol{spam}\ tweets}$$

Tweets	Total Tweets	Correctly Classified	Misclassified	Precision (%)	Recall (%)
All	7007	3815	3192	54.45	55.24
Only spam	**1993**	**1838**	**155**	**92.22**	**92.22**
Only non-spam	5014	2259	2755	45.05	-

Table 6.4: Spam Filter 2-Class Classification (Dataset 1)

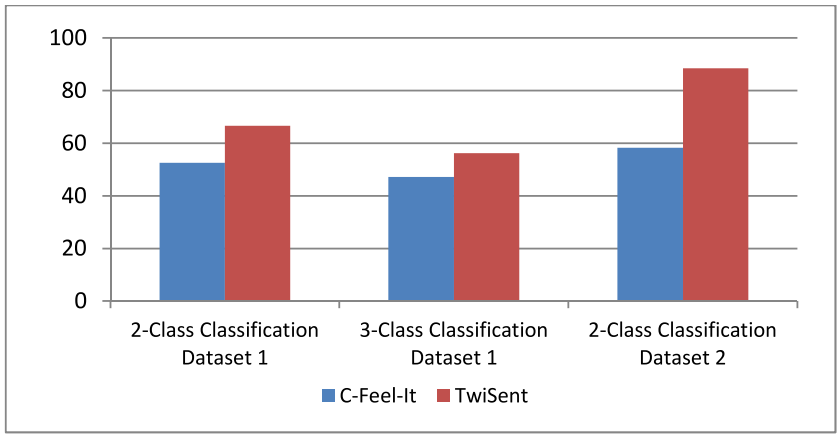
Tweets	Total Tweets	Correctly Classified	Misclassified	Precision (%)	Recall (%)
All	7007	5010	1997	**71.50**	**54.29**
Only spam	1993	1604	389	80.48	80.48
Only non-spam	5014	4227	787	84.30	-

Table 6.5: Spam Filter 4-Class Classification (Dataset 1)

The 4-class evaluation is done to see if a higher precision can be achieved. Having positive, negative and objective tweets in the same category is not a very good idea as they do represent completely different classes.

6.8.5 TwiSent Evaluation

For the overall system, we perform a 2-class and a 3-class classification (refer to *Table 6.6*) using TwiSent on *Dataset 1*. In the 2-class classification, we consider only *positive and negative* tweets, with all *objective* tweets ignored. In the 3-class setting, we consider *positive, negative* and *all objective* tweets as one separate class. Ablation tests (refer to *Table 6.8*) are performed by removing one module at a time and noting the resulting accuracy of the remaining system. This is done to find the sensitivity of each module. These tests are performed under the *2-class classification setting* using lexicon based classification. The confidence with which the accuracy changes are accepted to be statistically significant is also shown. *Graph 6.1* shows the accuracy comparison between *TwiSent* and *C-Feel-It* in datasets 1 and 2, under a 2-class and a 3-class classification setting.



Graph 6.1: Accuracy Comparison of TwiSent with C-Feel-It under Different Settings

System	2-class Accuracy	3-class Accuracy
C-Feel-It	52.58	47.23
TwiSent	**66.69**	**56.17**

System	2-class Accuracy	Precision/Recall

C-Feel-It	50.8	53.16/72.96
TwiSent	**68.19**	**64.92/69.37**

Table 6.6: Classification Accuracy of Lexicon based Approach for Classification - Dataset 1

Table 6.7: Classification Accuracy of the Supervised Approach for Classification (Dataset 1)

Module Removed	Accuracy	Statistical Significance Confidence (%)
Entity-Specificity	65.14	95
Spell-Checker	64.2	99
Pragmatics Handler	63.51	99
Complete System	**66.69**	**-**

Table 6.8: Results of Ablation Test using Lexicon-based Classification - Dataset 1

Supervised classification is performed using SVM (linear kernel with default parameters) under the 2-class classification setting using *leave-one-out* validation. 1200 samples from the positive and negative class each are used to train the classifier. *Table 6.7* shows the supervised classification accuracy.

TwiSent is evaluated on *Dataset 2* under a *2-class classification setting* where we consider only *positive* and *negative* classes. The performance of TwiSent as well as C-Feel-It on *Dataset 2* is presented in *Table 6.9*.

System	Positive Precision	Negative Precision	Overall Accuracy
C-Feel-It	69.06	48.2	58.24
TwiSent	**88.06**	**88.97**	**88.53**

Table 6.9: Classification Accuracy of Lexicon-based Approach for Classification - Dataset 2

6.9 DISCUSSIONS

6.9.1 Overall Accuracy

The Spam Filter module achieved a high precision (*92.22 %*) in a 2-class setting with only spams. Given a mixed bag of spam and non-spam tweets, its performance improved in a 4-class setting with an overall precision of *71.50%* as opposed to *54.45%* in case of a 2-class classification. This is because merging positive, negative and objective classes into a single class is undesirable as the 3 classes are unique and have different properties altogether. TwiSent achieved a much better accuracy over the baseline system under all the settings. In the 2-class setting the accuracy improvement is over *14%* whereas in the 3-class setting, it is around *10%*. TwiSent achieves a higher negative precision improvement than positive precision improvement (refer to *Table 7*) over C-Feel-It, which indicates it can capture negative sentiment strongly. The accuracy in the supervised system is low owing to the very sparse feature space due to inherent text limit of tweets.

6.9.2 Ablation Test

The accuracy changes after removing the Entity Specific module, Spell-Checker and Pragmatics Handler are *statistically significant at 95%, 99% and 99% confidence* respectively. The Ablation test shows that removing the Pragmatics Handler decreases the system accuracy most. This implies that Pragmatism is a very strong feature in the Social Media. The Spell-Checker also proved to be an important module owing to the tendency of people to mix and match shortenings and abbreviations which cannot be captured in standard lexicons. Hence, without this module, any lexicon-based system would miss out on many important cue words. The entity-specific module, *though important conceptually*, do not contribute greatly because of lack of context due to very short length of tweets, where people express opinions directly to the point unlike in reviews or blogs. The accuracy also gets affected due to the incorrect dependency relations given by the parser owing to the noisy text (mis-spelt words).

6.9.3 Effect of Artificially Created Training Samples

There has been a lot of work in Twitter that collect data based on specific features like *hashtags* (Alec *et al.*, 2009; Read *et al.*, 2005; Pak *et al.*, 2010)*, emoticons* (Gonzalez *et al.*, 2011) *etc*. and auto-annotate the tweets based on these features. Although these systems

achieve a very high accuracy, they remain biased towards these special features. In this work, we showed that although a system may work very well on a dataset based on a specialized feature set with hashtags (*Dataset 2*), it does not necessarily work well in a general setting (*Dataset 1*). This is evident in the performance of TwiSent in *Dataset 2* (created based on *hashtags*) where it attains a high accuracy of *88.53%* compared to the overall accuracy of *66.69%* in *Dataset 1* (manually annotated general purpose data). This shows that the specialized set of features used to crawl the data actually give away the sentiment *explicitly*, unlike in the general dataset which may have latent sentiment based out of *sarcasm, jokes, teasers* and other *implicit* sentiment, which is quite difficult to detect.

6.10 SUMMARY

We introduced a Twitter based multi-stage, sentiment analysis system, *TwiSent*, with specialized modules to tackle the nuances of micro-blogging genres. Our results suggest that we outperform a similar Twitter based sentiment application by *14%*. One of the major contributions of our work is in introducing Twitter based spams in the context of sentiment analysis. Our *Spam Filter* performs well not only as a part of the system but also as a stand-alone application. Apart from this module, we also showed that the *Spell-Checker* helps in text correction and normalization, whereas the *Pragmatics Handler* can loosely capture the pragmatics in text which assists in improving the classification performance. The *Entity-Specific* module helps in capturing sentiment pertaining to the search entity. The system cannot capture *sarcasm* or *implicit sentiment* due to the usage of a generic lexicon in the final stage for classification.

Overall, the work not only highlights the issues associated with the micro-blogs but also presents an effective system to handle many of them. We also show that a superlative system performance on a dataset collected and annotated using a specialized feature set does not guaranty a similar or comparable performance on real-life micro-blog data under a general setting.

Chapter 7

7. WIKISENT: A WEAKLY SUPERVISED SENTIMENT ANALYSIS SYSTEM USING EXTRACTIVE SUMMARIZATION WITH WIKIPEDIA

This chapter discusses the effect of external knowledge in sentiment analysis of reviews and proposes a computational model to incorporate world information in a sentiment analysis system with the help of Wikipedia. The roadmap for the rest of the section is as follows: We give an analysis of the various aspects of a movie review in *Section 7.1*. There we highlight the significant and non-significant concepts for sentiment analysis of movie reviews. *Section 7.2* describes how the ontological and sectional information in Wikipedia help in this task. Wikipedia is used to extract the movie-specific or genre-specific features comprising of the *movie plot*, *actors, directors* and other *film crew*, the *fictional characters* in the movie and *general movie domain specific features*. All these features are used to obtain an *extractive summary* of the review consisting of the reviewer's opinions only about the movie in focus. *Section 7.3* gives the algorithm for the extraction of the opinion summary, consisting of the *relevant* reviewer statements, which is classified by a sentiment lexicon in *Section 7.4*. The experimental evaluation is presented in *Section 7.5* on a gold standard dataset of 2000 movie reviews as well as on an unlabeled pool of 27,000 documents to find the trend. *Section 7.6* discusses the results followed by conclusions in *Section 7.7*.

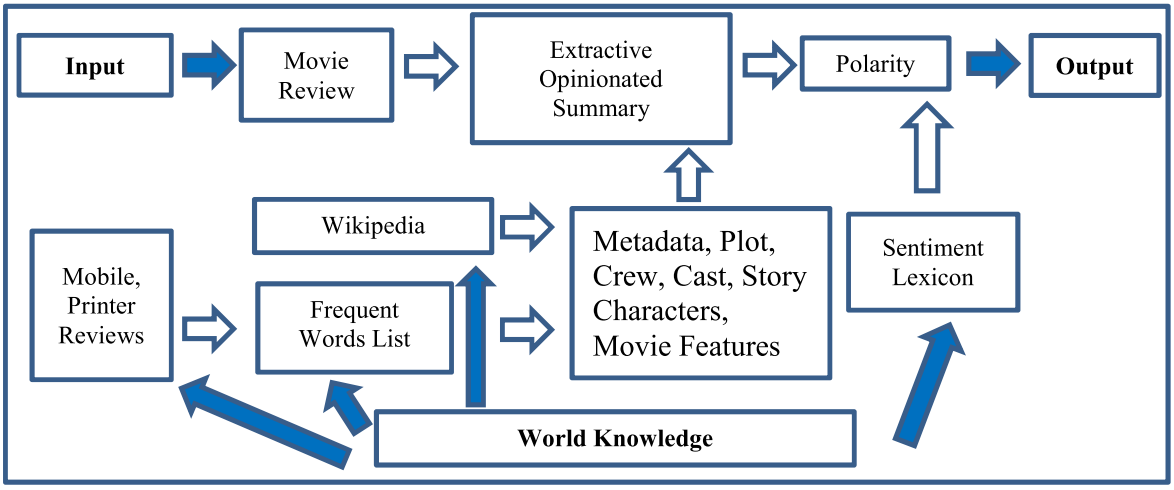


Figure 7.1: System Block Diagram

Consider the fragment of a review of the movie L.I.E taken from the IMDB movie review corpus (Pang *et al.*, 2002) which has been tagged as a negative review:

> "1. Best remembered for his understated performance as Dr. Hannibal Lecter in Michael Mann's forensics thriller, Manhunter, Scottish character actor Brian Cox brings something special to every movie he works on. 2. Usually playing a bit role in some studio schlock (he dies halfway through The Long Kiss Goodnight), he's only occasionally given something meaty and substantial to do. 3. If you want to see some brilliant acting, check out his work as a dogged police inspector opposite Frances McDormand in Ken Loach's Hidden Agenda.
>
> 4. Cox plays the role of Big John Harrigan in the disturbing new indie flick L*.I.E.*, which Lot 47 picked up at Sundance when other distributors were scared to budge. 5. Big John feels the love that dares not speak its name, but he expresses it through seeking out adolescents and bringing them back to his pad. 6. What bothered some audience members was the presentation of Big John in an oddly empathetic light. 7. He's an even-tempered, funny, robust old man who actually listens to the kids' problems (as opposed to their parents and friends, both caught up in the high-wire act of their own confused lives.). 8. He'll have sex-for-pay with them only after an elaborate courtship, charming them with temptations from the grown-up world" …..
>
> 9. It's typical of unimaginative cinema to wrap things up with a bullet, sparing the writers from actually having to come up with a complex, philosophical note. 10. In this regard, l*.i.e.* (and countless other indie films) share something in common with blockbuster action films: problems are solved when the obstacle is removed. 11. How often does real life work this way? To extend the question: if a movie is striving for realism , do dramatic contrivances destroy the illusion ?"

Example 7.1: Review of the Movie L.I.E

The first paragraph of the review talks about the central character Brian Cox's notable performance in some earlier movie. The second paragraph gives a brief description of his character in an empathetic light which comprises of positive opinions about the character. The reviewer opinion about the movie comes only in the last paragraph, where he gives some negative opinions about the movie. The review consists of majority positive words, not all of which are significant to the reviewer opinion, outweighing the negative opinions about the movie. A bag-of-words classifier, thus, would wrongly classify this review as positive.

7.1 FACETS OF A MOVIE REVIEW

Movie review analysis is a challenging domain in Sentiment Analysis (Turney, 2002) due to sarcasm, thwarting and requirement of extensive world knowledge. The reviewer opinion about the movie may target the characters in the movie, the plot or his expectations from the crew involved. We categorize the reviewer statements in nine categories in *Table 7.1*. We define *Crew* in a movie as the people who are involved in making of the movie like the *Producer, Director, Actor, Story-Writer, Cinematographer, Musician etc*. We are mainly interested in extracting opinions from *Category 8* (*Table 7.1*) where all the other Categories may lend a supporting role to back his opinions or add noise. We give examples (taken from movie reviews of the IMDB corpus - Pang *et al.*, 2002) for each of the categories.

General perception about the crew	*John Travolta is considered by many to be a* **one-hit wonder**.
Objective facts about the crew and movie	*Born into a family of thespians Kate Winslet came into her* **talent** *at an early age.*
Past performance of the crew	*The role that made Winslet an* **international star** *was Rose DeWitt Bukater, in James Cameron's Titanic (1997).*
Expectations from the movie or crew	*I cancelled the date with my girlfriend just to watch my* **favorite star** *in this movie.*
Movie plot	*Filmmaker Michael Cuesta uses it as a metaphor of* **dangerous** *escape for his 15-year old protagonist.*
Opinion about the characters in the movie	*He's an* **even-tempered, funny, robust** *old man who actually listens to the kids'* **problems**.
Characteristics of a movie or genre	*There is an axiom that directors having a* **big hit** *in debut have a* **big bomb** *with their second film.*
Opinion about the movie and crew	*While the movie is* **brutal**, *the* **violence** *is neither* **very graphic nor gratuitous**.
Unrelated category	*My grandson gave me passes to this new picture because the free screening is the same night as that* **horrible** *show ...*

Table 7.1: Reviewer Statement Categories with Examples

It is evident from the examples above why movie review text is difficult to analyze. Every example has some *opinion words* (shown in bold), not all of which are related to the movie in focus. A unigram based bag-of-words model would capture a lot of noise, if it considers all words to be equally relevant.

7.2 WIKIPEDIA ONTOLOGICAL INFORMATION EXTRACTION FOR MOVIE REVIEW ANALYSIS

Wikipedia is the largest English knowledge repository consisting of more than 21 million articles collaboratively written by volunteers all around the world. It is open-access and regularly updated by about 100,000 active contributors daily[24]. Each Wikipedia page is an article on some concept or topic. Each article belongs to one of the many defined categories and subcategories. For example, Category Film has 31 sub-categories like *Film making, Works about Films, Film Culture etc*. Furthermore, each article has a number of sections which can be edited separately. Any Wikipedia article on films may consist of sections like *Plot, Cast, Production, Marketing, Release etc*. which are common among most of the articles of that category. We utilize this feature to extract movie specific information from the Wikipedia.

A Wikipedia movie article consists of an info-box, just below the title of the article, giving information about the movie story and achievements in short. We call this section the *Metadata* of the movie. There is another info-box on the extreme right of the article which provides information about the name of the *Producer*, *Director, Actors, Cinematographer etc*. We call this section the *Crew* Information. There is an info-box on the movie plot which summarizes the story of the movie and talks about its fictional aspects. We call this section the *Plot* of the movie. There is another info-box, called the *Cast*, which gives information about the actors in the movie, the roles they perform and the name of characters they enact. We call this section the *Character* of the movie. We use all the above information extracted from the Wikipedia article, about the particular movie, to incorporate World Knowledge into the system.

7.2.1 Wikipedia Article Retrieval

The IMDB corpus (Pang *et al.*, 2002) has 2000 tagged reviews and 27,000 unlabeled ones. The tagged review documents had their titles removed. Thus the corresponding reviews had to

[24] http://en.wikipedia.org/wiki/Wikipedia

be retrieved from the unprocessed html documents and their titles had to be extracted. The title of the movie review was used to construct a *http get* request to retrieve the corresponding Wikipedia article on the movie. In case of multiple articles in different domains with the same name, the *film* tag was used to retrieve the desired article. For multiple movies with the same name, the year (in which the movie was released) information available with the title was used to extract the correct Wikipedia article. Thus the Wikipedia article retrieval was in the order *film name* → *film tag* → *film year*.

7.2.2 Metadata Extraction

The Metadata text was POS-tagged using a part of speech tagger[25] and all the *Nouns* were extracted. The Nouns were further stemmed[26] and added to the **Metadata** list. The words were stemmed so that *acting* and *action* have the same entry corresponding to *act*. Some movie articles in Wikipedia had the Plot section missing. The metadata was used in those cases to replace the Plot. In other cases, the *Metadata* information was simply appended to the *Plot* information.

7.2.3 Plot Extraction

The words in the movie Plot section were extracted similarly as in Metadata extraction. In both the Plot and Metadata, the concepts were restricted to be *Nouns*. For example, in the Harry Potter movie the Nouns *wizard, witch, magic, wand, death-eater, power etc*. depict concepts in the movie.

Considering all the Nouns incorporate a lot of noise into the **Plot** list as commonly used Nouns like *vision, flight, inform, way etc*. are added as well. To prevent this, both in the *Metadata* and the *Plot*, a separate list was created comprising of the frequently found terms (it will be shortly discussed how this list was compiled) in a corpus. Subsequently, the frequently occurring words were filtered out, leaving only *movie specific* concepts in the Metadata and Plot lists.

7.2.4 Why are only Nouns considered?

Only Nouns are considered to restrict the movie or genre-specific concepts to be entities. If Verbs are considered, a lot of noisy features would be incorporated in the list; as verbs are

[25] http://nlp.stanford.edu/software/tagger.shtml
[26] http://sourceforge.net/projects/stemmers/files/Lovins-stemmer-Java/

used in a wide variety of senses as well as with different subjects, not all of which are of interest. For example, consider "*Harry acted as if nothing has happened*" vs. "*Kate Winslet acted awesome in the movie.*" Here *act* is present as a Verb in both the sentences, where the first sentence belongs to the *plot* (*Category 5*) and the second sentence depicts the reviewer opinion (*Category 8*). The difference lies in the presence of different subjects of interest with the Verb. Our focus has been to capture the *subjects* and *objects* in the sentence which give direct clues about the *category* of the reviewer statement, so that the feature lists are as pure as possible.

7.2.5 Crew and Character Extraction

All the crew information was added to the **Crew** list. The character names were extracted and added to the **Character** list. These depict the fictional roles the actors play in the movie.

7.2.6 Frequent Words List Construction

The underlying hypothesis for this list creation is that movie reviews will have certain concepts and terms those are exclusive to this domain and will less frequently occur in other domains. Review data from the *Printer* and *Mobile Phone* domains[27] was used to create a list of frequently occurring terms in those domains. Since those domains are completely disjoint from the movie review domain, words which *frequently* occur in all of these domains should be commonly occurring words. Thus the commonly used words list consists of the frequently occurring terms in all these domains. The *tf-idf* measure was used and all those words above a threshold were added to the **FreqWords** list. For example, the word *person* (which is a Noun) occurred in all the domains with a very high frequency and was thus added to **FreqWords** list.

7.2.7 Domain Specific Feature List Construction

Wikipedia articles on films and aspects of films[28] were extracted. The sentences in those documents were POS-tagged. The Nouns were retrieved and frequently occurring words were removed. The remaining words were stemmed and added to the **MovieFeature** list. *Table 7.2* shows a snapshot of the genre specific terms automatically extracted from the Wikipedia article about movies.

[27]http://mllab.csa.iisc.ernet.in/downloads/reviewmining/fulldata.tar.gz
[28] http://en.wikibooks.org/wiki/Movie_making_manual

Movie, Staffing, casting, Writing, Theory, Writing, Rewriting, Screenplay, Format, Treatments, Scriptments, Synopsis, Logline, Pitching, Certification,scripts, Budget, Ideas, Funding, budgeting, Funding, Plans, Grants, Pitching, Tax, Contracts, law, Copyright, Pre-production, Budgeting, Scheduling, Pre-production, film , stock, Story, boarding, plot, Casting , Directors, Location, Scouting, …..

Table 7.2: Extracted Movie Domain Specific Terms

7.3 ALGORITHM TO EXTRACT OPINION SUMMARY

Section 7.3 describes the creation of the feature lists *Metadata, Plot, Crew, Character, FreqWords* and *MovieFeature*. Now, given a movie review the objective is to extract all those sentences that reflect the opinion of the reviewer *only* about the movie. This forms the *OpinionSummary* of the movie. A sentence-by-sentence analysis of the review is performed.

Any sentence not involving any word from any of the above lists is not considered relevant at all, thus pertaining to the *Unrelated Category 9*. Concepts from the *Plot, Metadata, and Character lists* are considered least significant, as they talk about the movie story and not about the reviewer opinion. They play a dampening role in extracting the reviewer's opinion about the movie. This covers *Category 5 and 6*. Concepts from the *MovieFeature* list are likely to comment on some specific aspect of the movie and are thus considered more relevant (*Category 7*). Finally, any concept from the movie *Title* or *Crew* list is considered *most relevant*, as the reviewer is likely to express his opinion about the movie. This covers *Category 1-4* and *8*. *The final opinion of the reviewer (Category 8) is actually a weighted function of all these concepts from different lists.*

The *Metadata, Plot and Character* lists are combined into a single list called the *Plot*. We now have 3 main categories of features corresponding to the *Plot, Crew* and *MovieFeature* lists with an auxiliary *FreqWords* list.

Given a movie review R with n sentences S_i, our objective is to determine whether each sentence S_i is to be *accepted* or *rejected* based on its *relevance* in judging the reviewer opinion. Let each sentence S_i consist of n_i words $w_{ij}, j \in 1 \ldots n_i$. The *Plot* list does not consist of any word from the *FreqWords* list or the *MovieFeature* list. Similarly, the *MovieFeature* list also does not contain any word from the *FreqWords* list. The *relevance factor* of the sentence S_i is given by,

$$Rel_{factor_i} = 2\sum_j 1_{w_{ij} \in Crew\ or\ MovieTitle} + \sum_j 1_{w_{ij} \in MovieFeature} - \sum_j 1_{w_{ij} \in Plot, w_{ij} \notin Crew, w_{ij} \notin MovieTitle}$$

Equation 7.1: Relevance Factor of a Sentence

The relevance factor is actually a weighted combination of the various *features* from the *lists*. It *counts* the words appearing from different lists in a sentence. The concepts belonging to the crew or movie title are *most important* as they are directly related to the movie in focus; hence any words appearing from those lists are given *double* the weight than concepts from the other lists. This is because any reviewer statement containing the name of the movie, actors, directors, story-writer *etc*. is likely to express his opinion directly about the movie in focus. The concepts belonging to the *plot* are not so much relevant in judging the reviewer's opinion about the movie and may add noise. They play a dampening role as they signify a concept related to the movie story and not the reviewer opinion, which is captured in the '-' sign.

Any sentence S_i is accepted if Acc_{factor_i} corresponding to S_i is 1.

$$Acc_{factor_i} = 1 \ \boldsymbol{if}\ Rel_{factor_i} \geq 0\ and\ \exists w_{ij} \in S_i \ \boldsymbol{s.t.}\ w_{ij} \in Crew\ or\ MovieFeature\ or\ Movie\ Title$$
$$= 0\ otherwise$$

Equation 7.2: Acceptance Factor of a Sentence

Any sentence is accepted as being relevant, if the number of *weighted significant concepts* from the movie *Title*, *Crew* and *MovieFeature* lists, are greater than or equal to the number of concepts from the less significant *Plot* and *Character* lists. This implies that Rel_{factor_i}should be greater than or equal to zero for a sentence to be accepted. It is equal to zero when the number of significant and less significant concepts in the sentence is equal.

Considering the movie review in *Example 7.1*, the algorithm works as follows:

In Sentence [1], only *Brian Cox* belongs to the *Cast* list and *actor* and *movie* belong to the *MovieFeature* list.

Rel_{factor_1}=2*1+1*2-1*0=4 ≥ 0, $Acc_{factor_1} = 1$ and the sentence is accepted. In [2], there is no keyword from the lists and it is rejected straightaway. [3] has the keyword *acting* from MovieFeature and it is accepted where, Rel_{factor_3}=1, $Acc_{factor_1} = 1$. [4] has the keywords *Cox, L.I.E* from *Cast and MovieTitle*, *John Harrigan* from *Character* list and *distributor* from the *MovieFeature* list. Rel_{factor_4}=2*2+1-1=4 ≥ 0 and the sentence is accepted. [5] has only

the keyword *Big John* from *Character* and the relevance factor. Rel_{factor_5}=0+0-1=-1$\not\geq$ 0 and the sentence is rejected. [6] has the keyword *audience* from *MovieFeature* and *Big John* from *Character*. Its Rel_{factor_6}=0+1-1=0 ≥ 0 and it is accepted. [7] has the keywords *temper, friend* from *Plot* and Rel_{factor_7}=0+0-2=-2 $\not\geq$ 0 and it is rejected. [8] has the keywords *sex, charm* from *Plot* and Rel_{factor_8}=0+0-2=-2 $\not\geq$ 0 and it is rejected. [9] has the keywords *cinema, writers* from *MovieFeature* and *bullet* from *Plot*. Thus Rel_{factor_9}=0+1*2-1=1 ≥ 0 and it is accepted as being relevant. [10] has the keywords *l.i.e* from *MovieTitle, films(2), action* from *MovieFeature*. Thus $Rel_{factor_{10}}$=2*1+1*3-0=5 ≥ 0 and it is accepted as being relevant. [11] has the keyword *movie* from *MovieFeature*. $Rel_{factor_{11}}$=0+1*1-0=1 ≥ 0 and it is accepted.

Input : Review R
Output: OpinionSummary

Step 1: Extract the Crew list from Wikipedia

Step 2: Extract the Plot list from Wikipedia

Step 3: Extract the MovieFeature list from Wikipedia

Step 4: Extract the FreqWords list as the common frequently occurring concepts in Mobile Phone, Printer and Movie domains.

Let $OpinionSummary = \emptyset$
for i=1..n
  if $Acc_{factor_i} == 1$
    add S_i *to* $OpinionSummary$
  end if
end for

Algorithm 7.1:. Extract Opinion Summary from Review

7.4 CLASSIFICATION OF THE OPINION SUMMARY

The words in the extracted opinion summary can be directly used as features in a supervised classification system. Since we do not use any labeled data for training, a sentiment lexicon is used in the final phase to classify the opinion summary. A sentiment lexicon contains an opinion word along with its polarity. SentiWordNet (Baccianella *et al.*, 2010), General Inquirer (GI) (Stone *et al.*, 1966) and Subjectivity (Wilson *et al.*, 2005) sentiment lexicons are

used to find the polarity of a word. SentiWordnet is tagged at the synset level (*word sense and polarity*) whereas the remaining lexicons contain the words and their most commonly considered polarity. While using the SentiWordNet, we use the *first sense* of a word as we do not perform word sense disambiguation. Negation handling is done in lexical classification. The polarity of any word in a window of 5, from the occurrence of any of the negation operators *not, neither, nor and no*, is flipped. The final polarity of the review (*pos* or *neg*) is given by a *majority voting of the polarity of the individual words* in the extracted opinion summary.

7.5 EVALUATION

The experimental evaluation is performed on the IMBD movie review corpus (Pang *et al.*, 2002). It consists of 2000 reviews collected from the IMDB movie review site and polarity labeled at the document level, 1000 from each class. This forms our gold standard data. Apart from this, there is an unlabeled corpus of 27,886 unprocessed html documents from which the above 2000 reviews have been extracted and labeled by the annotators.

7.5.1 WikiSent Evaluation on the Gold Standard Data

Three baselines are considered for WikiSent:

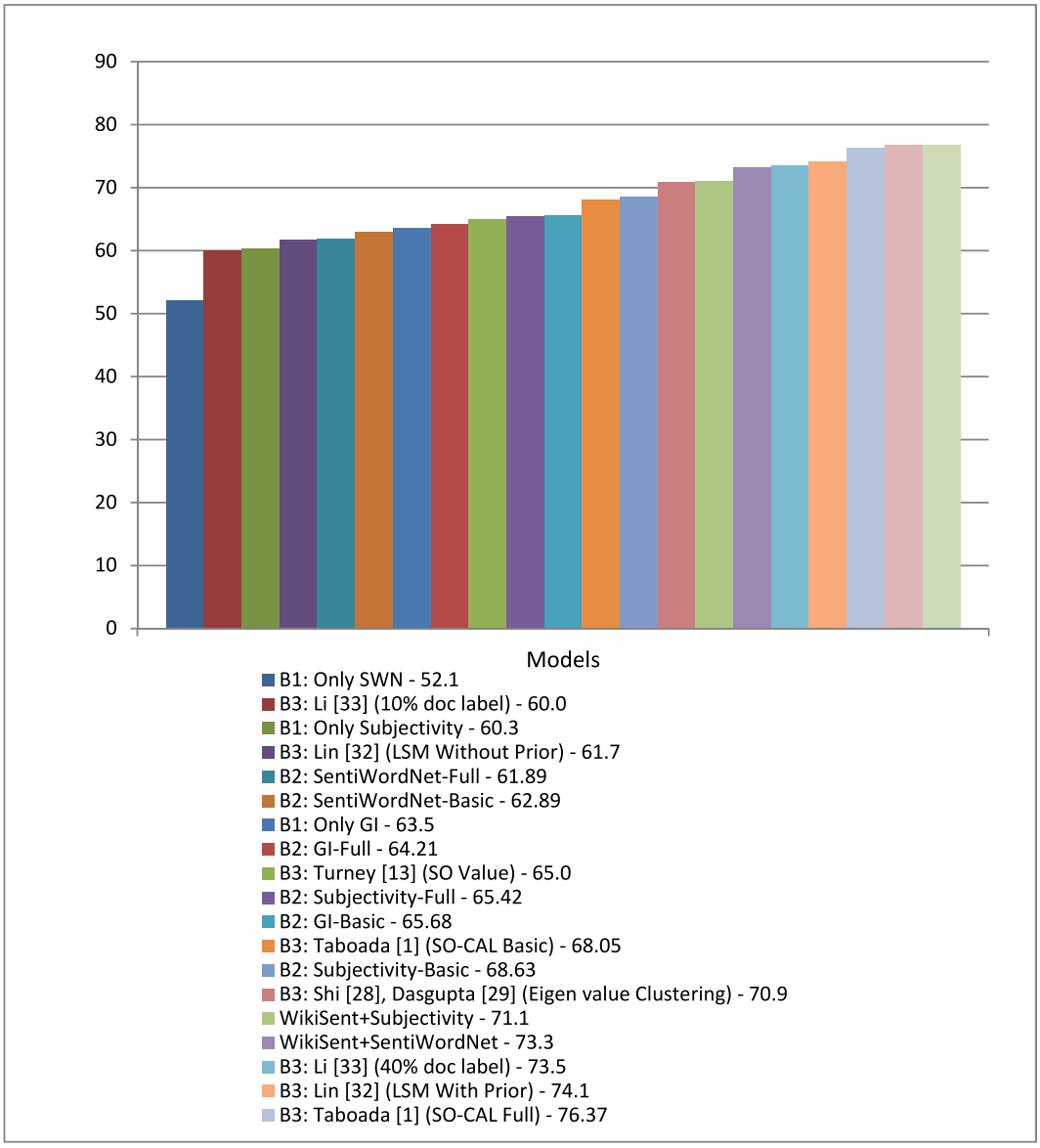
1) *Baseline 1* is taken as the bag-of-words lexicon based classification (majority voting on opinion words) using *SentiWordNet* (Baccianella *et al.*, 2010), *General Inquirer (GI)* (Stone *et al.*, 1966) and *Subjectivity* (Wilson *et al.*, 2005) lexicons.

2) *Baseline 2* is taken as the lexicon based approach of Taboada *et al.* (2011) to determine sentiment from text. The system, Semantic Orientation Calculator (SO-CAL), uses dictionaries of words annotated with semantic orientation (polarity and strength) and incorporates negation and intensification.

3) The best performing unsupervised, weakly supervised and semi-supervised systems in the movie review domain, on the same dataset, are taken as *Baseline 3*.

Baseline 2 has been chosen as a sophisticated version of the bag-of-words model (*Baseline 1*). WikiSent only handles negation; it *does not* handle intensification or strength of emotions, like SO-CAL. Part of the baseline accuracy *Table 7.3* is adapted from Taboada *et al.* (2011), in which the evaluation is done on the same dataset as ours, but with different sentiment lexicons. Three lexicons are chosen with which SO-CAL gives the *worse, median* and *best*

performance corresponding to the *SentiWordNet, General Inquirer and Subjectivity* lexicons. *Table 7.3* also shows the performance of *Baseline 1* and *WikiSent* using those three lexicons. *Table 7.4* shows accuracy variation of $Rel_{factor_i} \geq \boldsymbol{\theta}$. *Table 7.5* shows the accuracy comparison of WikiSent with the systems in *Baseline 3*. *Graph 7.1* shows the accuracy comparison of WikiSent with different systems in baselines 1, 2 and 3.



B1: Baseline 1, B2: Baseline 2, B3: Baseline 3

Graph 7.1: Accuracy Comparison of WikiSent with Different Systems

Baseline 1 : Simple Bag-of-Words	
Only SentiWordNet	52.1
Only Subjectivity	60.3
Only GI	63.5
Baseline 2 : Worse, Median and Best Performing Lexicons with SO-CAL	
SentiWordNet-Full	61.89
SentiWordNet-Basic	62.89
GI-Full	64.21
GI-Basic	65.68
Subjectivity-Full	65.42
Subjectivity-Basic	68.63
WikiSent with Different Lexicons	
WikiSent+Subjectivity	71.1
WikiSent+SentiWordNet	73.3
WikiSent+GI	**76.85**

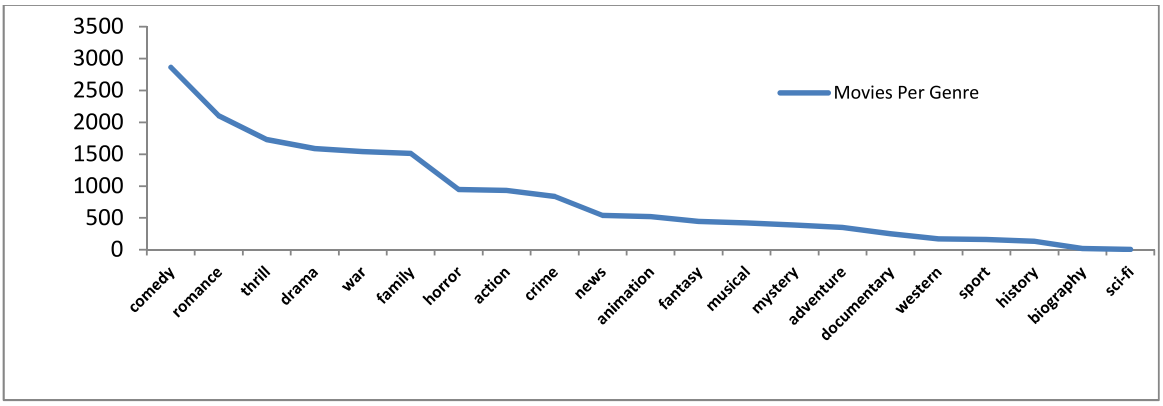
Table 7.3: Baseline System Accuracy Comparison with WikiSent

Systems	Classification Method	Accuracy
Turney SO value	Unsupervised (PMI)	65
Taboada SO-CAL Basic [1]	Lexicon Generation	68.05
Taboada SO-CAL Full [1]	Lexicon Generation	76.37
Shi [28], Dasgupta [29]	Unsupervised Eigen Vector Clustering	70.9
Socher [30] RAE	Semi Supervised Auto Encoders	76.8
Lin [32] LSM	Unsupervised without prior info	61.7
Lin [32] LSM	Weakly Supervised with prior info	74.1
Li [33]	Semi Supervised 10% doc. Label	60
Li [33]	Semi Supervised 40% doc. Label	60
WikiSent	**Wikipedia+GI Lexicon**	**76.85**

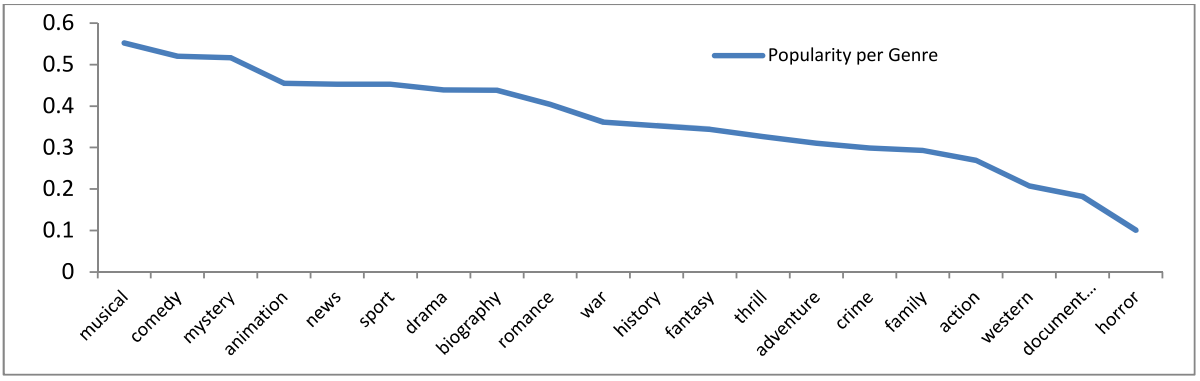
Table 7.4: Accuracy Comparison of WikiSent with Different Systems (Baseline 3)

7.5.2 Movie Trend Analysis using WikiSent

This analysis is performed on the unprocessed pool of 27,886 html documents. The movie reviews belong to 20 different genres. *Graph 7.1* shows the number of movies belonging to each genre in the dataset as well as all the genre names.



Graph 7.2: Movies per Genre in the Dataset



Graph 7.3: Genre Popularity in the Dataset

	WikiSent	Bag-of-Words Baseline 1
Positive Reviews (%)	48.95	81.2
Negative Reviews (%)	51.05	18.79

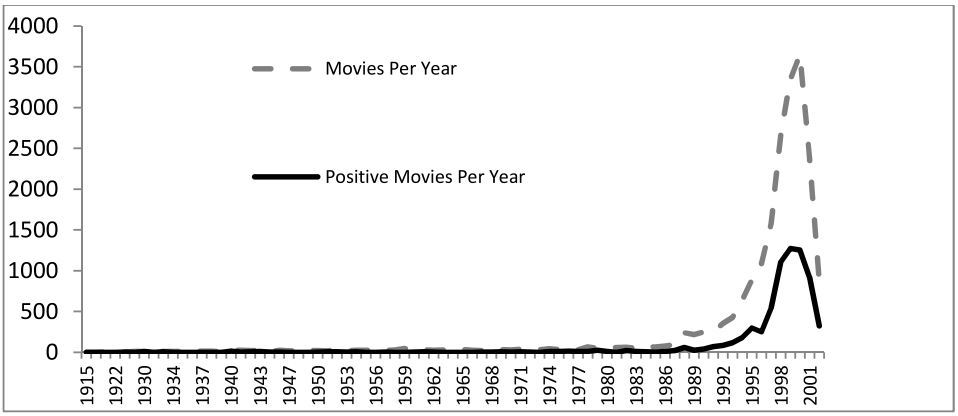
Table 7.5: Movie Review Polarity Comparison of WikiSent vs. Baseline System

Table 7.6 gives the fraction of the movie reviews that are predicted to be positive and negative by WikiSent and *Baseline 1*(expressed in *percentage*).

The genre popularity (refer to *Graph 7.2*) is given by:

$$Genre\ Popularity = \frac{Positive\ Movie\ Reviews\ per\ Genre}{Total\ Movie\ Reviews\ per\ Genre}$$

Graph 7.3 shows the total number of movies released and the total number of movies predicted to be positive *per year* (as is present in the dataset).



Graph 7.4: Movie Popularity per Year in the Dataset

7.6 DISCUSSIONS

7.6.1 WikiSent Performance Analysis

It is observed that *WikiSent* performs the best with *Inquirer* (GI) among all the other lexicons used with it. It is interesting to find the huge accuracy leap from the baseline accuracy using *SentiWordNet* (Baseline 1 – **52.1%**; Baseline 2- **61.89, 62.89**) and *SentiWordNet* + *WikiSent* (**73.3**) in *Table 7.3*. This accuracy improvement is achieved through the deployment of extractive summarization using Wikipedia, by which the objective sentences are eliminated from the review before classification. Using GI with WikiSent gives an accuracy improvement of 13% over *Baseline 1* and 11% over *Baseline 2*. The corresponding figures with the Subjectivity lexicon are 11% and 3% respectively.

The value of $\boldsymbol{\theta}$ (*Table 7.4*) has been taken as zero in *Equation 7.2*, without using any development set in accordance to the reasons given in *Section 7.4*. It is observed from *Table 7.4*, that it indeed happens to be the optimal value of a threshold for Rel_{factor_i} for accepting a sentence as significant *if* $Rel_{factor_i} \geq \boldsymbol{\theta}$. As the value of $\boldsymbol{\theta}$ increases, fewer sentences are considered relevant due to which many informative sentences are left out. Higher value of $\boldsymbol{\theta}$ means a single sentence should have a large number of representatives from the *Crew*, *MovieFeature* lists, which is rare. Again, as $\boldsymbol{\theta}$ decreases more number of sentences are considered relevant which incorporate noise due to the inclusion of many insignificant sentences. Low value of $\boldsymbol{\theta}$ means the number of insignificant features from the *Character, Plot* lists may outnumber those from *Crew*, *MovieFeature* lists.

WikiSent achieves a better accuracy than all the existing unsupervised, weakly supervised and semi-supervised systems on the same dataset, as is evident from *Table 7.5*. It is also notable that WikiSent is able to beat the semi-supervised systems, which use partial document labels, without using *any* labeled training data.

7.6.2 Movie Trend Analysis

The movie review corpus contains most movies from the genres *comedy, romance, thrill, drama, war, horror, action, crime* and least number of movies from the genres *west, sport, history, biography, sci-fi* (in the descending order). This depicts a general trend in the movie making of sticking to the most popular genres.

It is observed that movies belonging to the categories *musical, comedy, mystery, animation* and *news* received the most number of positive reviews whereas movies belonging to the genres *family, action, western, documentary* and *horror* received the least number of positive reviews. This shows that although there are a large number of movies from the *comedy* genre, in general these movies tend to do well; whereas the movies from the *action and horror* genres, despite having a large number of releases, do not fare very well. The movies from the genres *musical and animation*, generally have a good acceptance despite less number of releases.

The number of movies per year has grown exponentially with time as is observed from *Graph 7.3*. This also highlights the importance of movie review analysis in the socio-economic aspect. It is seen that the number of movies as well as good movies have increased with time. The dip after the year 2000 may be attributed to the fact that the data collection process was only till 2002, so the reviews crawled after the year 2000 were less.

The number of negative reviews in the movie domain actually outweighs the number of positive reviews, to some extent (refer to *Table 7.6*). This shows that people are a bit skeptical in calling a movie *good*. It is also seen that the baseline bag-of-words model, which tags a *huge* number of movie reviews as positive, is unable to catch this trend. This also shows the limitation of the baseline model which considers all words to be relevant, in analyzing movie reviews.

7.7 WIKISENT DRAWBACKS AND ERROR ANALYSIS

One of the major drawbacks of the system is that it does not perform co-reference resolution due to which valuable information is lost. Thus any sentence having a feature anaphorically

referring to a relevant feature in the previous sentence will be ignored, due to which relevant sentences may be rejected as *false negatives*, during extractive summarization. A sentence may also be rejected as non-significant if it contains the *synonymous variation* of a concept from the feature lists. Usage of a lexical dictionary like the WordNet will alleviate this problem. But this error occurs much less as most of the genre-specific concepts in the *MovieFeature* list (like *acting, direction, script etc*.) occur in the reviews in the same lexical form. Although there may be some synonymous *Plot* concepts in Wikipedia and the IMDB review, it has been observed that the significant movie-specific concepts (Nouns) occur in both of them in the same lexical form. For example, the concepts *wand, death-eater, muggle, wizard etc*. in the Harry Potter IMDB review will occur in the same form in Wiki, rather than in synonymous form.

A sentence may be accepted as a *false positive,* during extract creation, if it refers to some previous movie of the film crew or state his general perceptions about them, which are not directly related to the current movie. False negatives are comparatively easier to deal with, but false positives pose a problem as they may not be completely irrelevant from the cognitive point of view of the reviewer. This is because past experience of the reviewer with the film crew may bias his perception about the current movie.

We do not perform word-sense disambiguation[29], in this work. Since we consider only the first sense of a word, we miss out on its actual sense and its proper polarity in some cases (McCarthy *et al.*, 2004). For example we use a simple lexicon which does not distinguish between the various meanings of the same word, like '*bank*' in the sense of '*relying'* which is a *positive term* and '*bank*' in the sense of a '*river bank*' which is *objective*. Furthermore the lexicon, that gives the maximum accuracy with WikiSent, has a low coverage. If WikiSent is used with all the SO-CAL features (Ex:*strength of emotions*) and a dictionary with high coverage more accuracy improvement would be possible. *Inquirer* suffers from a low coverage since it is manually hand-tagged. Many of the polarity-bearing words are absent in it. Though SWN has a large coverage, it is highly biased towards positive and objective terms.

7.8 SUMMARY

In this work, we proposed a weakly supervised approach to sentiment classification of movie reviews. The polarity of the review was determined by filtering out irrelevant objective text

[29]http://en.wikipedia.org/wiki/Word-sense_disambiguation

from relevant subjective opinions *about the movie in focus*. The relevant opinionated text extraction was done using Wikipedia.

We showed that the incorporation of world knowledge through Wikipedia, to filter out irrelevant objective text, can significantly improve accuracy in sentiment classification. Our approach differs from other existing approaches to sentiment classification using Wikipedia in the fact that *WikiSent* does not require any labeled data for training. Weak supervision comes from the usage of resources like POS-tagger and Sentiment Lexicons. This work is different in the way it creates an *extractive opinionated summary* of the review using the *ontological* and *sectional* information from Wikipedia and then uses a lexicon to find its polarity. The work analyzes the significance of the various aspect specific features in movie reviews that are relevant to sentiment analysis and harnesses this information using Wikipedia.

The system suffers from the lack of handling *synonymous concepts*, *anaphora resolution* and *word sense disambiguation*. Usage of a simple lexicon at the final stage for polarity calculation also mars its accuracy.

Nevertheless, we showed that our system attains a *better accuracy* than all the existing unsupervised, weakly supervised and semi-supervised systems in the domain on the same dataset, without using any labeled data for training. In addition to this, we also do a general analysis of the movie review corpus, widely used in sentiment analysis, using WikiSent (*based on the genre, year of release, movie review polarity*) to show the general trends persisting in movie-making as well as public acceptance of the movie.

Chapter 8

8. YOUCAT: A WEAKLY SUPERVISED SYSTEM FOR YOUTUBE VIDEO CATEGORIZATION FROM USER COMMENTS AND META DATA USING WORDNET AND WIKIPEDIA

This chapter proposes a weakly supervised system for Youtube video categorization that incorporates external knowledge through WordNet and Wikipedia and social content through the user posts. The chapter is organized as follows: *Section 8.1* discusses the unsupervised feature extraction from various sources. *Section 8.2* gives the algorithm for feature vector classification and genre identification. The experimental evaluations are presented in *Section 8.3* followed by discussions of the results in *Section 8.4* and conclusions in *Section 8.5*.

8.1 FEATURE CONSTRUCTION

A seed list of words is created for each genre by searching a *thesaurus*. A concept list is created for each genre with *relevant words* from the *WordNet* and *named entities* in *Wikipedia* with the help of the seed list of the corresponding genre.

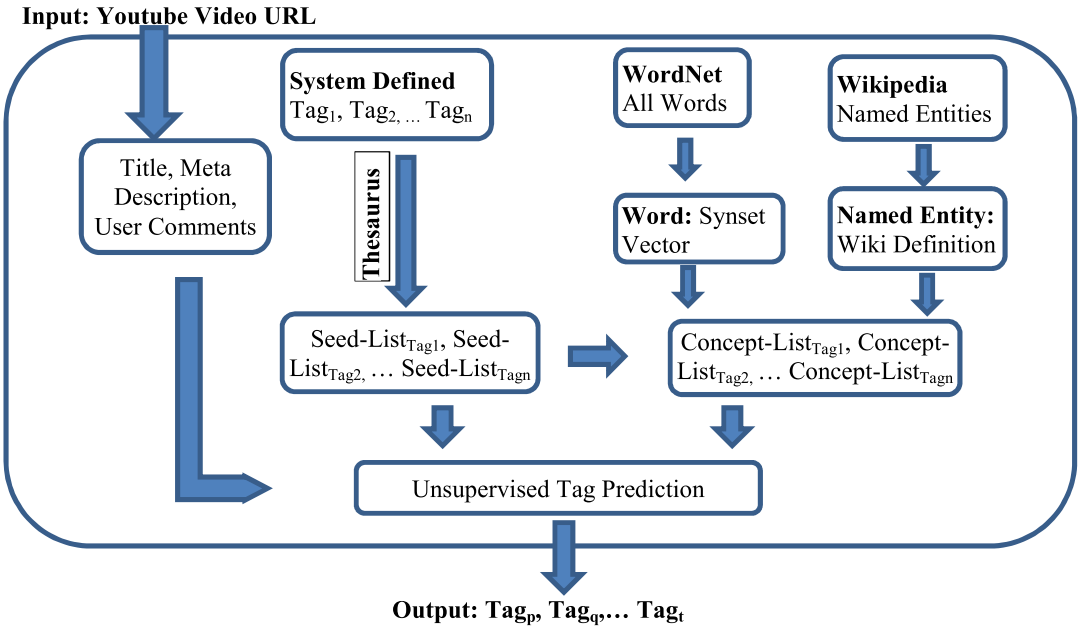


Figure 8.1: System Block Diagram

Given a video descriptor consisting of the *video title*, the *meta-description of the video* and the *user comments*, the seed list and the concept list for each genre is used for finding appropriate matches in the *video descriptor* to predict appropriate tags or categories for the video.

8.1.1 Data Pre-Processing

A set of tags is defined in the system, beforehand, for which a seed list is created. All the entries in the WordNet[30] and the *named entities* in Wikipedia[31] with their corresponding *definitions* are hashed and stored. The seed list creation and hashing are done, beforehand, to facilitate efficient query and retrieval during real-time genre prediction.

8.1.2 Seed List Creation

A *seed list* of words is created for each genre (*defined in the system*) which captures the key characteristics of that category. For Example, "*love", "hug", "cuddle" etc.* are the characteristics of the *Romance* genre. A *root word* or the *name of the genre* is taken and all its synonyms are retrieved from a thesaurus. For example, the word *Laugh* is taken for its genre *Comedy* and all its first level synonyms are retrieved which are again recursively used to retrieve their level-one synonyms, till a list of size 50-60 words is compiled. A thesaurus is used for this purpose which gives every day words and slangs. In our work, the following thesaurus[32] retrieves the words *rofl, roflmao, lol etc.* when the word *Laugh* is looked up from the *Comedy* genre. A *snapshot* of the seed list is shown in *Table 8.1* with the genre name.

Comedy (25)	funny, humor, hilarious, joke, comedy, roflmao, laugh, lol, rofl, roflmao, joke, giggle, haha, prank
Horror (37)	horror, curse, ghost, scary, zombie, terror, fear, shock, evil, devil, creepy, monster, hell, blood, dead, demon
Romance (21)	love, romantic, dating, kiss, relationships, heart, hug, sex, cuddle, snug, smooch, crush, making out
Sports (35)	football, game, soccer, basketball, cheerleading, sports, baseball, FIFA, swimming, chess, cricket, shot
Tech (42)	internet, computers, apple, iPhone, phone, pc, laptop, mac, iPad, online, google, mac, laptop, XBOX, Yahoo

Table 8.1: Snapshot of Seed List for Each Genre

[30] wordnetweb.princeton.edu/perl/webwn
[31] www.wikipedia.org
[32] www.urbandictionary.com/thesaurus.php

8.1.3 Concept Hashing

Each word (used as a *key* for hashing) in the WordNet, that is not present in any seed list, is hashed with the set of all its synsets. A synset is a set of synonyms that collectively disambiguate each other and give a unique sense to the set. For example, the word *dunk* is stored in the hashtable with its set of synsets - {*dunk, dunk shot, stuff shot*; *dunk*, *dip, souse, plunge, douse*; *dunk*; *dunk, dip*}. Technically, we should have taken only the words in the synset of its most appropriate sense. Since we do not perform word sense disambiguation and taking only the first sense provides less context while classifying the feature vector, the information from all the senses of a given word are used. In a way, we are trying for the closest sense match.

Wikipedia is necessary for *named entity* expansion, since the WordNet does not contain most of these entries. All the *named entities* in Wikipedia with the *top 2 line definition* in their corresponding Wiki articles are stored in a hashtable. For example, *NBA* is stored in the hashtable with its definition from the Wikipedia article as {*The National Basketball Association (NBA) is the pre-eminent men's professional basketball league in North America. It consists of thirty franchised member clubs, of which twenty-nine are located in the United States and one in Canada.*}.

8.1.4 Concept List Creation

Let *w* be any given word and its expanded form given by WordNet (*set of all its synsets*) or Wikipedia (*top 2 line definition*) be denoted by w'. Let w'_j be the j[th] word in the expanded word vector. Let $seed_k$ be the seed list corresponding to the k^{th} genre. The genre of *w* is given by

$$genre(w) = argmax_k \sum_{j:\, w'_j \in seed_k} \mathbf{1}$$

Equation 8.1: Concept List Creation

Here, $\mathbf{1}$ is an indicator function which returns 1 if a particular word is present in the seed list corresponding to a specific genre and 0 otherwise. In the given example, with the pre-defined 5 genres (*Table 8.1*), *dunk* and *basketball* both will be classified to the *Sports* genre as they have the maximum matches ("*shot", "basketball"*) from the seed list corresponding to the *Sports* genre in their expanded concept vector.

Finally, a *concept list* is created for each genre containing *associated* words in the WordNet (ignoring those in the seed lists) and named entities in the Wikipedia.

8.1.5 Video Descriptor Extraction

Given a video url the *video title*, the *meta description* of the video and all the *user comments* on the video from Youtube are retrieved. A *stopwords* list is used to remove the words like *is, are, been etc*. A lemmatizer[33] is used to reduce each word to its base form or lemma. Thus "*play"*, "*played*", "*plays*", "*playing*" are reduced to its lemma "*play*".

Consider the sentence in a video descriptor, "*It was an awesome slam dunk in the NBA finals by Michael Jordan*". None of the words in this sentence is present in any seed list. But *dunk* and *NBA* are present in the concept list corresponding to the *Sports* genre and thus the given sentence is associated to *Sports*. The association (*Sports* via *Basketball*) can also be captured by considering the named entity *Michael Jordon* in Wikipedia.

8.2 FEATURE VECTOR CLASSIFICATION

Let the video descriptor '*f*' consist of '*n'* words, in which the j^{th} word is denoted by '$word_j$'. The *seed list* and the *concept list* for the k^{th} genre are denoted by $seed_k$ and $concept_k$ respectively. The score of f belonging to a particular $genre_k$ is given by,

$$score(f \in genre_k; w_1, w_2) = w_1 \times \sum_{j:word_j \in seed_k} \mathbf{1} + w_2 \times \sum_{j:word_j \in concept_k} \mathbf{1}$$

Equation 8.2: Feature Genre Scoring

Here, $\mathbf{1}$ is an indicator function that returns 1 if a word is present in the seed list or concept list corresponding to $genre_k$and 0 otherwise. Weights w_1 *and* w_2 are assigned to words present in the seed list and the concept list respectively $where\ w_2 < w_1$. This is to reduce the effect of *topic drift* during concept expansion. The score of a video belonging to a particular genre is,

$$score(video \in genre_k; p_1, p_2, p_3) =$$
$$p_1 \times score\left(f^{Title} \in genre_k\right) + p_2 \times score(f^{Meta\ Data} \in genre_k) +$$
$$p_3 \times score(f^{Comments} \in genre_k)$$

Equation 8.3: Video Genre Scoring

Here p_1, p_2, p_3 denote the weight of the feature belonging to the *title, meta data (meta description of the video)* and *user comments* respectively where $p_1 \geq p_2 > p_3$. This is to assign more importance to the title, then the meta data and finally the user comments. Finally the genre of the video is given by,

[33] Dragon.nlp.tool.lemmaister.EngLemmatiser

$$video_{genre} = argmax_k\ score(video \in genre_k)$$

Equation 8.4: Single Genre Prediction

This assigns the highest scoring genre as the predicted category for the video. But most of the popular videos in Youtube can be attributed to more than one *genre*. Thus to allow multiple tags to be assigned to a video, a thesholding is done and the prediction is modified as:

$$video_{genre} = k, if\ score(video \in genre_k) \geq \theta$$

$$where\ \theta = \frac{1}{k}\sum_k score(video \in genre_k)$$

Equation 8.5: Multiple Genre Prediction

If the score of the video for any genre is greater than the average score of all the genres, then it is assigned as a possible tag for the video. In case the genre scores for the 5 categories are something like {*400, 200, 100, 50, 10*} with *avg=152*, then the first 2 genres are chosen. If any of the genre score is very high compared to the others, the average will rise decreasing the chance of other genres being chosen.

Pre-processing:

1. *Define Genres*
2. *Create a seed list for each genre by breadth-first-search in a Thesaurus, using a root word in the genre or the genre name*
3. *Create a concept list for each genre using all the words in WordNet (not present in seed lists) and named entities in Wikipedia using Equation 8.1*

Input: Youtube Video Url

1. *Extract Title, Meta Description of the video and User Comments from Youtube to form the video descriptor*
2. *Lemmatize all the words in the descriptor removing stop words*
3. *Use Equations 8.2-8.4 for genre identification of the given video*

Output: Genre Tags

Algorithm 8.1: Genre Identification of a Youtube Video

Algorithm 1 describes the genre identification steps in short. Since all the required lists are maintained as a hashtable, the genre of any word can be found in O(1) time complexity. Thus the time complexity of the genre identification algorithm is O(|W|), where |W| is the number of words in the video descriptor.

8.3 EVALUATION

The following 5 genres are used for evaluation: *Comedy, Horror, Sports, Romance and Information Technology.* 1398 videos are crawled from the Youtube with about 280 videos from each of the 5 genres. *Table 8.2* shows the number of videos from each genre. Seed words like "*comedy*", "*ghost*", "*sports*", "*love*" and "*computer*" are used as keywords for finding the videos with the keywords in the *titles* or *uploader assigned tags*. Subsequently the keywords are *removed* from the video title and meta-data to remove the bias.

Romance	Comedy	Horror	Sports	Tech
252	300	300	263	283

Table 8.2: Number of Videos in Each Genre

Only the 1st page of user comments are taken with those comments less than 150 characters in length. The user comments are normalized by removing all the punctuations and reducing words like "*loveeee*" to "*love*". The number of user comments varied from 0 to 800 for different videos. *Table 8.3* shows the average number of user comments for the videos in each genre.

Romance	Comedy	Horror	Sports	Tech
103.64	168.94	83.4	56.91	223.45

Table 8.3: Average User Comments for Each Genre

The first integer values satisfying the constraints in the equations are taken as parameter values, which are set as $w_1 = 2, w_2 = 1, p_1 = p_2 = 2 \; and \; p_3 = 1$.

Support Vector Machines Classifier[34] with various features, like combination of unigrams and bigrams, incorporating part-of-speech (POS) information, removing stop words, using lemmatization *etc.*, is taken as the baseline. The words in the video descriptor consisting of the title, meta-description of the video and the user comments are taken as features for the SVM. A *linear kernel* is used with *10-fold* cross validation. *Table 8.4* shows the baseline system accuracy with various features. SVM with lemmatized unigrams and bigrams as features, ignoring stop words, gave the maximum accuracy of 84.36% .

The system does not tag every video with a genre. It will not tag a video if it does not find a clue in the video descriptor that is present in the seed list or the concept list (*i.e.* the scores are all *zero*); or when there are ties with scores for multiple genres being equal. The precision, recall and f_1-score for each genre are defined as:

[34] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

$$precision = \frac{number\ of\ videos\ correctly\ tagged}{number\ of\ videos\ tagged} \times 100$$

$$recall = \frac{number\ of\ video\ correctly\ tagged}{number\ of\ videos\ present\ in\ the\ genre} \times 100$$

$$f_1\ score = \frac{2 * precision * recall}{precision + recall}$$

SVM Features	F_1-Score(%)
All Unigrams	82.5116
Unigrams+Without stop words	83.5131
Unigrams+ Without stop words +Lemmatization	83.8131
Unigrams+Without stop words +Lemmatization+ POS Tags	83.8213
Top Unigrams+Without stop words +Lemmatization+POS Tags	84.0524
All Bigrams	74.2681
Unigrams+Bigrams+Without stop words+Lemmatization	**84.3606**

Table 8.4: SVM Baseline with Different Features

Table 8.5 shows the performance of YouCat for single genre prediction (*Equation 8.4*) with precision, recall and f_1-score for 2 cases: (1) Prediction is done without using user comments and without the help of Wikipedia or WordNet and (2) Prediction is done using user comments but without the help of Wikipedia and WordNet. *Table 8.6* shows the performance of YouCat when concept expansion is introduced in the system though Wikipedia and WordNet.

Genre	Without User Comments + Without Wikipedia & WordNet			With User Comments + Without Wikipedia & WordNet		
	Precision	Recall	F_1-Score	Precision	Recall	F_1-Score
Romance	76.26	66.27	70.91	77.36	74.60	**75.95**
Comedy	43.96	40.00	41.89	69.23	66.00	**67.58**
Horror	80.47	68.67	74.10	76.45	70.33	**73.26**
Sports	84.21	68.71	75.67	85.07	69.94	**76.77**
Tech	90.83	73.50	81.25	84.09	78.45	**81.17**

Table 8.5: Single Genre Identification with and without User Comments, without using Wikipedia & WordNet

Genre	Without User Comments + With Wikipedia & WordNet			With User Comments + With Wikipedia & WordNet		
	Precision	Recall	F_1-Score	Precision	Recall	F_1-Score
Romance	76.06	70.63	73.24	80.16	78.57	**79.36**
Comedy	47.31	44.00	45.6	77.08	74.00	**75.51**
Horror	75.63	70.33	72.88	75.78	73.00	**74.36**
Sports	87.5	73.01	79.60	89.05	74.85	**81.33**
Tech	92.34	85.16	88.60	92.03	89.75	**90.88**

Table 8.6: Single Genre Identification with and without User Comments, using Wikipedia &WordNet

Table 8.7 shows the performance of YouCat when the system is allowed to predict multiple genres for a given video (*Equation 8.5*). The prediction is taken to be correct if the originally labeled tag is one of the predicted tags. This is necessary as often multiple genres are associated with a video.

Genre	Without User Comments + With Wikipedia & WordNet			With User Comments + With Wikipedia & WordNet		
	Precision	Recall	F_1-Score	Precision	Recall	F_1-Score
Romance	86.97	82.14	84.49	94.78	93.65	**94.21**
Comedy	77.5	73.67	75.54	91.78	89.33	**90.54**
Horror	83.92	80.00	81.91	89.56	88.67	**89.11**
Sports	96.38	81.6	88.38	96.38	81.6	**88.38**
Tech	96.63	92.34	94.44	98.56	92.03	**95.18**

Table 8.7: Multiple Genre Identification with and without User Comments, using Wikipedia &WordNet

It may seem that the performance improvement for multiple genre identification, in our case, is trivial to achieve as the system can achieve 100% accuracy by simply assigning all the given genres (5 in our case) to a given video. This is because the prediction is taken to be correct if any of the predicted tags matches with the labeled tag. Thus an important performance measurement parameter is the number of predicted tags for each video. *Table 8.8*

shows the average number of predicted tags for each video in each genre, with and without user comments.

	Average Tags/Video Without User Comments	Average Tags/Video With User Comments
Romance	1.45	1.55
Comedy	1.67	1.80
Horror	1.38	1.87
Sports	1.36	1.40
Tech	1.29	1.40
Average	**1.43**	**1.60**

Table 8.8: Average Predicted Tags/Video in Each genre

Table 8.9 shows the confusion matrix when single genre identification is done with User Comments, Wikipedia &WordNet. The ijth entry of the matrix (row-major) is defined as:

$$Mat_{ij} = \frac{number\ of\ videos\ predicted\ as\ genre\ j}{number\ of\ videos\ labeled\ as\ genre\ i} \times 100$$

	Romance	Comedy	Horror	Sports	Tech
Romance	80.16	8.91	3.23	4.45	3.64
Comedy	3.13	77.08	3.47	9.03	7.29
Horror	10.03	9.34	75.78	3.46	1.38
Sports	0.70	7.30	0	89.05	2.92
Tech	0.72	5.07	0.36	1.81	92.03

Table 8.9: Confusion matrix for Single Genre Prediction

Table 8.10 shows the average f_1-score for the different models used.

Table 8.10: Average F_1-Score of Different Models

Graph 8.1 shows the incremental performance improvement for each genre due to the inclusion of *user comments*, *concept expansion* and *multiple tag prediction*.

8.4 DISCUSSIONS

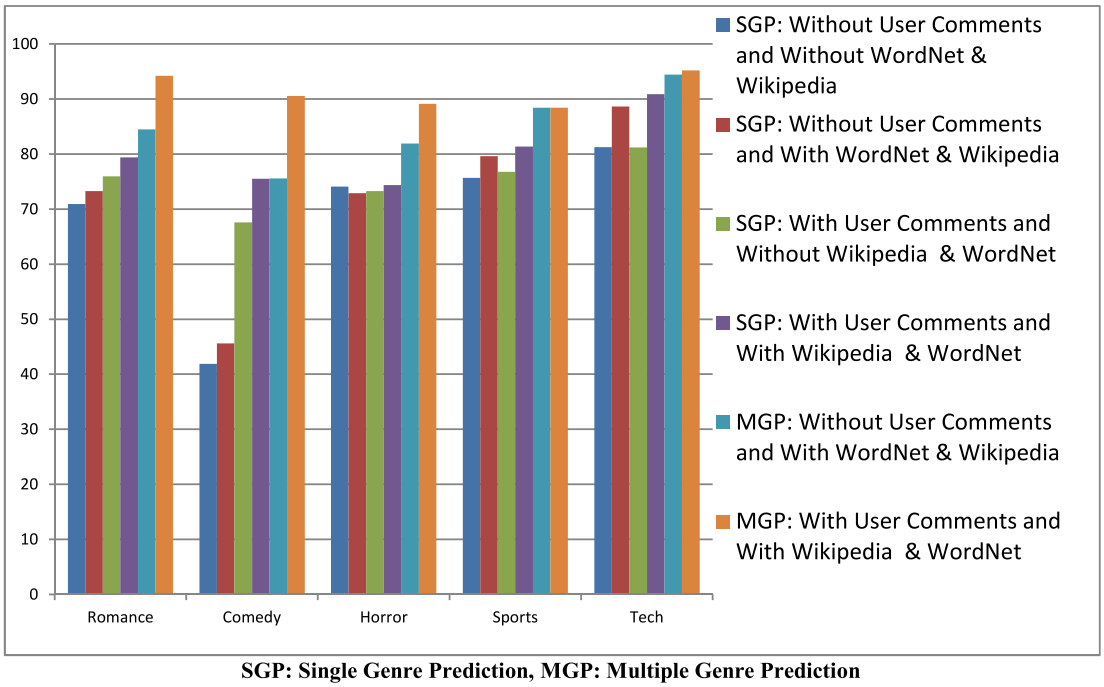
8.4.1 Multi-Class SVM Baseline

The SVM has been taken as the baseline as it is found to perform the best in text classification and video categorization works. Ignoring stop words in the feature vector improved the accuracy of SVM over the all-unigram feature space. Further accuracy improvement is

achieved by lemmatization. This is because all the related unigram features like *laugh, laughed, laughing etc*. are considered as a single entry *laugh,* which reduces the sparsity of the feature space.

Model	Average F_1 Score
SVM Baseline: Single Genre Prediction With User Comments	84.3606
Single Genre Prediction : Without User Comments + Without Wikipedia & WordNet	68.76
Single Genre Prediction : With User Comments + Without Wikipedia & WordNet	74.95
Single Genre Prediction : Without User Comments + With Wikipedia & WordNet	71.984
Single Genre Prediction : With User Comments+ With Wikipedia &WordNet	*80.9*
Multiple Genre Prediction : Without User Comments + With Wikipedia & WordNet	84.952
Multiple Genre Prediction : With User Comments + With Wikipedia & WordNet	**91.48**

Table 8.11: Accuracy Comparison of Different Models in YouCat



SGP: Single Genre Prediction, MGP: Multiple Genre Prediction

Graph 8.1: Genre-wise F-Score Improvement for Different Models

The part-of-speech information further increased accuracy, as they help in *crude* word sense disambiguation. Consider the word *haunt* which has a noun synset and gloss as {*haunt,*

hangout, resort, repair, stamping ground -- (a frequently visited place)}. It also has 3 verb synsets where the first verb sense is {*haunt, stalk -- (follow stealthily or recur constantly and spontaneously to; "her ex-boyfriend stalked her"; "the ghost of her mother haunted her")*}. Using POS information, the word haunt will have two entries now corresponding to *Noun_haunt* and *Verb_haunt*. Although the second sense is related to the *Horror* genre, the first sense is not which can only be differentiated using the POS tags.

Top unigrams help in pruning the feature space and removing the noise which helps in accuracy improvement. The top unigrams include the seeds words used for each genre in our model (*Table 8.10*).

Using *only bigrams* however decreases the accuracy as many unrelated pairs are captured which do not capture the domain characteristics. Using bigrams along with unigrams gives the highest accuracy. This is because the entities like *Michael Jordon* can be used as features as a whole, unlike in unigrams.

8.4.2 Overall Accuracy

Though our system could not beat the multi-class SVM baseline of *84.36%* in single genre prediction, it nevertheless achieved a descent f_1-score of *80.9%*, *without using any labeled data for training.* The multiple genre prediction, however, beats the baseline with *91.48%* f_1-score.

8.4.3 Effect of User Comments

The user comments often introduce noise through the off-topic conversations, spams, abuses *etc.*; the slangs, abbreviations and pragmatics prevalent in the user posts make proper analysis difficult. For this reason, less weight is given to the user comments than to the video meta-description. However, an improvement of *6 percentage point* and *9 percentage point* in the f_1 score for single genre prediction (without and with concept expansion respectively) using the comments, suggest that the greater context provided by the user comments provide more clues about the genre to help in genre identification. The corresponding improvement in the multiple genre prediction using concept expansion is around *7 percentage point*.

When concept expansion is not used, user comments contribute a performance improvement of *5 percentage point* in Romance, *1 percentage point* in Sports and a huge *26 percentage point* in Comedy. This suggests that the user information mostly helps in identifying funny videos, as well as romantic videos to some extent. Horror videos undergo mild performance degradation by incorporating user comments. Using concept expansion,

user comments contribute an accuracy improvement of *6 percentage point* in Romance, a huge *30 percentage point* in Comedy and *2 percentage point* in the other genres.

8.4.4 Effect of Concept Expansion

Wikipedia and WordNet have been used in quite a number of works for concept expansion. In the genre identification task, using a seed set for each genre runs the risk of *topic drift.* This may occur as a concept may be identified to belong to an incorrect genre due to off-topic words by considering a larger context. However, less weight is given to concept expansion than to a direct match in the seed list to alleviate this risk. In single genre prediction using concept expansion, an f_1-score improvement of 3 *percentage point* (when user comments are not used) and 6 *percentage point* (when user comments are used) show that Wikipedia and WordNet help in associating unknown concepts to their respective genres with the help of lexical and world knowledge.

When user comments are not used, concept expansion contributes a performance improvement of *3 percentage point* in Romance, *4 percentage point* in Comedy, Sports and 7 *percentage point* in Tech. This suggests that the external knowledge sources help in easy identification of new technological concepts. Horror videos, again, undergo mild performance degradation. Using the comments, concept expansion contributes an accuracy improvement of *8 percentage point* in Comedy and over *9 percentage point* in Tech. Again, the performance improvement in Comedy using Wikipedia can be attributed to the identification of the concepts like *Rotfl, Lolz, Lmfao etc*.

8.4.5 Average Number of Tags per Video in Multiple Genre Prediction

The number of predicted tags in multiple genre identification for each video, on an average, is *1.43* and *1.6* in the two cases (without and with user comments). This suggests that mostly a single tag and in certain cases bi-tags are assigned to the video. It is also observed that the average number of tags per video increases when user comments are used. This is due to the greater contextual information available from user comments leading to genre overlap.

8.4.6 Confusion between Genres

The confusion matrix indicates that Romantic videos are frequently tagged as Comedy. This is often because many Romantic movies or videos have light-hearted Comedy in them, which is identifiable from the user comments. The Horror videos are frequently confused to be Comedy, as users frequently find them funny and not very scary. Both Sports and Tech videos

are sometimes tagged as Comedy. The bias towards Comedy often arises out of the off-topic conversation between the users in the posts from the *jokes, teasing etc*.

Overall, from the precision figures, it seems that Sports and Tech videos are easy to distinguish from the remaining genres. This is because the concepts in these two domains are frequently *monosemous* than in the other domains.

8.4.7 Issues

Many named entities in the Youtube media, especially unigrams, are ambiguous. Incorrect concept definition retrieval from the Wikipedia, arising out of ambiguity may inject noise into the system or can be ignored. For Example, a Sports video with the title "*Manchester rocks*" refers to the *Manchester United Football Club*. But Wikipedia returns a general article on the *city of Manchester* in England. None of the words in its definition matches any word in seed word lists and the entity is ignored.

Considering only WordNet synsets gives less coverage. Considering the gloss information helps to some extent. For example, if the word "*shot*" is not present in the seed list for *Sports*, then "*dunk*" cannot be associated to the *Sports* genre. But this association can be properly captured through the gloss of the WordNet first sense of "*dunk*" (- *a basketball shot in which the basketball is propelled downward into the basket*). However, it runs the risk of incorporating noise. Consider the word *good* and the gloss of one of its synsets {*dear, good, near -- with or in a close or intimate relationship*}. Here the word "*good*" is associated to *Romance* due to the presence of "*relationship*", which is incorrect.

Uploader provided video meta-data is typically small and require concept expansion to extract useful information. User comments provide a lot of information but incorporate noise as well. Auto-generated bot advertisements for products, off-topic conversation between users, fake urls, mis-spelt words, different forms of slangs and abbreviations mar the accuracy. For example, an important seed word for the *Romance* genre will not be recognized if "*love*" is spelt as "*luv*", which is common.

8.5 SUMMARY

In this work, we propose a completely unsupervised system, *YouCat*, for predicting *possible genre tags* for a video using the *video title, meta description* and the *user comments*. *Wikipedia* and *WordNet* are used for expanding the extracted concepts to detect cue words from a genre-specific seed set of words. There are a number of parameters which have been simplistically set. Tuning the parameters using labeled data may improve the accuracy. An

accuracy of *80.9%* in single genre prediction and *91.48%* in multiple genre prediction is obtained without using *any labeled data,* compared to the supervised SVM baseline of *84.36%* for single genre prediction. The accuracy suffers due to the inherent noise in the Youtube media arising out of the user comments and incorrect concept expansion due to ambiguous words.

This work is significant as it does not use any manually labeled data or supervision and is more than just a keyword search. It can be automatically extended for multiple genres without supervision. This work also exhibits the usefulness of user information and concept expansion though WordNet and Wikipedia in video categorization.

Chapter 9

9. LEVERAGING SENTIMENT TO COMPUTE WORD SIMILARITY

The underlying hypothesis that we follow for creating this metric is that *knowing the sentiment is beneficial in measuring the similarity*. In order to implement a metric based on this hypothesis, we incorporate sentiment values of the words being compared. Similar to Wan & Angryk (2007), we follow WordNet gloss based comparison technique to develop this metric. Gloss based technique has an inherent advantage over edge-based and information content-based metrics as it is applicable to all POS categories without any distinction. The rest of this chapter is organized as follows: We describe our metric and related terminologies in *Section 9.1*. We explain evaluation strategy in *Section 9.2*. *Section 9.3* describes existing similarity metrics that we use for comparison. Experimental setup is given in *Section 9.4*. Results of the experiments are discussed and analyzed in *Section 9.5*.

9.1 SENSIM METRIC

9.1.1 Gloss vector

We represent gloss of a synset in the form of a vector, we define this vector as gloss vector. To obtain the gloss of the words being compared, the corresponding sense used for each word needs to be known. We assume that we are provided with the synset corresponding to each word that needs to be compared. In other scenarios where the synset corresponding to word are not given, a close approximation can be taken by using respective WordNet first sense. Each dimension of the gloss vector represents a sentiment score of the respective content word. To obtain the sentiment scores, we use an external sentiment lexicon and assign sentiment values based on different scoring functions.

We use SentiWordNet 1.0[1] as the external sentiment lexicon to incorporate the sentiment values in the gloss vector. This Wordnet based resource has polarity scores attached to synsets (Esuli & Sebastiani 2006). Each synset in this resource is marked with 3 scores: a positive score, a negative score and an objective score, with the scores summing up to 1. As the sentiment scores are attached to synsets rather than lexemes, we disambiguate the

WordNet gloss to obtain the corresponding synsets. Based on synsets thus found, we assign sentiment scores to each dimension of the gloss vector.

Representing gloss in the form of vectors is not new, but novelty of our approach is in the incorporation of sentiment score to each dimension of the gloss vector.

9.1.2 Augmenting gloss vector

Gloss contains few content words averaging between 5-7. This creates a sparse vector space. To reduce the sparseness of the gloss vector, we augment the original gloss with the gloss of the *related synsets*. We use different WordNet relations, based on the POS category, to find the related synsets. Apart from the *adjacent* related synset, we add more context by further expanding the related synsets using synsets of content words of the original gloss.

Not all WordNet relations can be used for the expansion procedure as degree of sentiment content may change or not get carried to the next level. By taking relative transfer of the sentiment content from one synset to another for different WordNet relations, we empirically found a set of WordNet relations suitable for each POS category. Details of the WordNet relations used for expansion process are given in *Table 9.1*

POS	WordNet relations used for expansion
Nouns	*hypernym, hyponym, nominalization*
Verbs	*nominalization, hypernym, hyponym*
Adjectives	*also see, nominalization, attribute*
Adverbs	*derived*

Table 9.1: WordNet Relations used for Enhancing the Context of Gloss Vector

9.1.3 Scoring function

As SentiWordNet provides 3 scores for each synset: positive score, negative score and objective score, we devise a scoring function to capture the sentiment content of the words as a single real value. We explain four variants of the scoring function used in this study below:

9.1.3.1 Sentiment Difference (SD)

Difference between the positive score and the negative score is taken as the sentiment score of the synset concerned. Here SWNpos(A) signifies the positive score pertaining to the synset A. The sign of the score represents the orientation of the sentiment. If the sign is negative, the word has a negative connotation otherwise it has a positive connotation. If the value is zero, it is objective in nature.

$$Score_{SD}(A) = SWN_{pos}(A) - SWN_{neg}(A)$$

Equation 9.1: Sentiment Difference

9.1.3.2 Sentiment Max (SM)

For this function, we use the greater of the positive or negative score of the synset as sentiment score of the synset concerned.

$$Score_{SM}(A) = \max(SWN_{pos}(A), SWN_{neg}(A))$$

Equation 9.2: Sentiment Max

The orientation of the word is again distinguished by the sign. For negative connotation, a negative of the score returned is used else a positive score is taken.

9.1.3.3 Sentiment Threshold Difference (TD)

As the gloss is represented in the form of a vector with each dimension representing a sentiment score, dimensions which have zero magnitude may not contribute to sentiment content but the presence of each dimension is necessary for overall similarity computation. In order to avoid such scenarios, we introduce a threshold value that ensures, in case of objective words, a zero value is never encountered. We take a threshold value of 1 to compute the variant of SD scoring function.

9.1.3.4 Sentiment Threshold Max (TM)

SM scoring function is modified to handle zero magnitude problem as explained above.

$$Score_{TM}(A) = sign(\max(SWN_{pos}(A), SWN_{neg}(A))) * (1+abs(\max(SWN_{pos}(A), SWN_{neg}(A)))$$

Equation 9.3: Sentiment Threshold Max

9.1.4 Computing similarity

To compute the similarity between two word pairs, we take the cosine similarity of their corresponding gloss vectors.

$$SenSim_x(A, B) = cosine\ (gloss_{vec}(\text{sense}(A)),\ gloss_{vec}(\text{sense}(B)))$$

Where,

$gloss_{vec}$ $=1{:}score_x(1)\ 2{:}score_x(2)\ldots\ n{:}score_x(n)$

$score_x(Y)$ = Sentiment score of word Y using scoring function x

x = Scoring function of type SD/SM/TD/TM

Equation 9.4: Computing Cosine Similarity between Gloss Vectors

9.2 EVALUATION

To evaluate SenSim, we follow two methodologies: an intrinsic evaluation and an extrinsic evaluation. For intrinsic evaluation, we compare the correlation of the metric with a gold standard dataset. We also compare correlation of different existing similarity metrics with this gold standard and show how our new metric performs. To perform an extrinsic evaluation, we use SenSim metric to mitigate the effect of the unknown feature problem in supervised sentiment classification. Details of the same are explained in the following subsections.

9.2.1 Intrinsic evaluation: correlation with human annotators

We develop a new similarity dataset and manually annotate them with similarity scores. We did not use existing similarity datasets as we require the correct sense of the words being compared.

Dataset for annotation task

We chose 48 random word pairs for this task with each POS category having 12 word pairs each. Sentences, containing these words, are constructed to get the exact sense of these word pairs. Based on these sentences, words are sense disambiguated using WordNet 2.1 to get the corresponding synsets. A part of the word pairs used in this work is given in *Table 9.2.* Each word pair has a WordNet synset offset, prefixed with the POS, category attached to it.

Annotation strategy

To test our hypothesis that *incorporation of sentiment into a similarity metric gives better*

result than comparing similarity based on meaning alone, we perform a similarity annotation task between two human annotators. The annotators were asked to annotate word pairs using two different strategies.

Word Pairs	
regular_42374008	*accustomed_426235*
tardily_3100974	*lately_3108293*
randomly_371128	*specifically_341621*
pretentious_41915502	*arrogant_41957189*
defense_1811665	*attack_1958708*

Table 9.2: A Section of Dataset used for Experiments

- Annotation based on Meaning: Instruction were given to each annotator to give a score between 1-5 to word pairs based on semantic similarity with a score of 5 representing synonymous word pairs and 1 representing no relation between the words.
- Annotation based on Sentiment and Meaning combined: In this case, the annotators were asked to rate on a scale between 1-5 whether words were interchangeable given similar context and similar sentiment content. A score of 5 implies perfect interchangeability.

The dataset generated thus forms our gold standard data. Correlation between the annotator score is taken to test the hypothesis. As a part of this evaluation, we also take the correlation scores between different existing similarity metrics with the gold standard data, and compare the respective correlation scores with SenSim.

9.2.2 Extrinsic evaluation: synset replacement using similarity metrics

In this section, we evaluate SenSim metric using an application of similarity metrics, and compare the application performance with that of widely used similarity metrics. We use the synset replacement strategy using similarity metrics by Balamurali et al. (2011) for evaluating our metric. In this study, the authors showed that similarity metrics can be used to mitigate the effect of unseen feature problem in supervised classification. The objective of their study is to classify documents based on their sentiment content into positive class or negative class. Each

document to be classified is represented as a group of synsets obtained after sense disambiguation of the content words of the document.

Input: Training Corpus, Test Corpus,
Similarity Metric
Output: New Test Corpus
T:= Training Corpus;
X:= Test Corpus;
S:= Similarity metric;
train concept list = *get list concept*(T);
test concept list = *get list concept*(X);
for *each concept C in test concept list* do
temp max similarity = 0;
temp concept = C;
for *each concept D in train concept list* do
similarity value = get similarity value(C,D,S);
if *(similarity value > temp max similarity)* then
temp max similarity= similarity value;
temp concept = D;
end
end
C = temp concept;
replace synset corpus(C,X);
end

Return X;

Algorithm 9.1: Synset Replacement using Similarity Metric

A document level sentiment classifier is created based on the training corpus and its performance measured is on the test corpus. A trained classifier will not perform with same accuracy if new features are found in the test corpus. To mitigate this effect, they follow a *replacement strategy*. In this strategy, if a synset encountered in a test document is not found in the training corpus, it is replaced by one of the synsets present in the training corpus. The

substitute synset is determined on the basis of its similarity with the concerned synset in the test document. The synset that is replaced is referred to as an *unseen synset* as it is not known to the trained model.

Algorithm 9.1 shows the replacement algorithm devised by Balamurali et al. (2011). The algorithm follows from the fact that the similarity value for a synset with itself is maximum. Similarity metrics used in this study are explained in the following section.

Annotation Strategy	Overall	NOUN	VERB	ADJECTIVES	ADVERBS
Meaning	0.768	0.803	0.750	0.527	0.759
Meaning + Sentiment	0.799	0.750	0.889	0.720	0.844

Table 9.3: Pearson Correlation Coefficient between Two Annotators for Various Annotation Strategies

9.3 METRICS USED FOR COMPARISON

We compare SenSim with three existing similarity metric. They are:

LIN: The metric by Lin (1998) uses the information content individually possessed by two concepts in addition to that shared by them. The information content shared by two concepts A and B is given by their most specific subsumer (lowest superordinate(*lso*)). Thus, this metric defines the similarity between two concepts as

$$sim_{LIN}(A,B) = \frac{2 \times \log Pr(lso(A,B))}{\log Pr(A) + \log Pr(B)}$$

Equation 9.5: Lin Similarity

Lesk: Each concept in WordNet is defined through gloss. To compute the Lesk similarity (Banerjee & Pedersen 2002) between A and B, a scoring function based on the overlap of words in their individual glosses is used.

Leacock and Chodorow (LCH): To measure similarity between two concepts A and B, Leacock & Chodorow (1998) compute the shortest path through hypernymy relation between them under the constraint that there exists such a path. The final value is computed by scaling the path length by the overall taxonomy depth (D).

9.4 EXPERIMENTAL SETUP

Word sense disambiguation of WordNet glosses is carried out using the WSD engine by Zhong & Ng (2010). It is an all-words generic WSD engine with an accuracy of 82% on standard WSD corpus. We use WordNet::Similarity 2.05 package by Pedersen et al. (2004) for computing the similarity by other metric scores mentioned in this work. We use Pearson correlation coefficient to find the inter-annotator agreement.

For evaluation based on synset replacement using similarity metrics, we use the dataset provided by Balamurali et al. (2011). The experiments are performed using C-SVM (linear kernel with default parameters[3]), using bag-of-synsets as features, available as a part of LibSVM[4] package. All classification results reported are average of five-fold cross-validation accuracies.

To evaluate the result, we use accuracy, recall and precision as the metrics. Classification accuracy defines the ratio of the number of true instances to the total number of instances. Recall is calculated as a ratio of the true instances found to the total number of false positives and true positives. Precision is defined as the number of true instances divided by the number of true positives and false negatives. Positive Precision (PP) and Positive Recall (PR) are precision and recall for positive documents while Negative Precision (NP) and Negative Recall (NR) are precision and recall for negative documents.

Metric Used	Overall	NOUN	VERB	ADJECTIVES	ADVERBS
LESK (Banerjee *et al.*, 2003)	0.22	0.51	-0.91	0.19	0.37
LIN (Lin, 1998)	0.27	0.24	0.00	NA	Na
LCH (Leacock *et al.*, 1998)	0.36	0.34	0.44	NA	NA
SenSim (SD)	0.46	0.73	0.55	0.08	0.76
SenSim (SM)	**0.50**	0.62	0.48	0.06	0.54
SenSim (TD)	0.45	0.73	0.55	0.08	0.59
SenSim (TM)	0.48	0.62	0.48	0.06	0.78

Table 9.4: Pearson Correlation(r) of Various Metrics with Gold Standard Data

9.5 RESULTS AND DISCUSSION

9.5.1 Sentiment as a parameter for finding similarity

Table 9.3 shows correlation scores between two annotators for different annotation strategies. Apart from the correlation of the complete word pairs, it also shows POSwise correlation.

From the results, it is clearly evident that similarity is best captured when sentiment is also included. The annotation strategy involving a combination of meaning and sentiment has a better correlation among annotators (0.799) than the one which considers meaning alone. This verifies the hypothesis that taking sentiment content of words being compared is beneficial in assessing the similarity between them.

A POSwise analysis of *Table 9.3* suggests that apart from the Noun category, all other categories have a better correlation among annotators in assessing the sentiment based similarity annotation strategy. This may be due to the fact that in case of word pairs belonging to the Noun category, sentiment does not play much role. The highest correlation is seen for Verbs. In case of Adjectives, annotators have a fairly high correlation. Since most of the Adjective are sentiment bearing words, annotators might have found easier to compare them. In summary, a similarity metric which incorporates sentiments may be more beneficial to POS categories other than Nouns.

Table 9.4 shows the correlation between scores obtained using different similarity metrics with the scores obtained from gold standard dataset of an annotator. The correlation with respect to different POS categories is also shown. NA in some of the columns represent word pairs, belonging to those POS categories, which cannot be handled using the similarity metric concerned. For example, metrics like LIN and LCH cannot handle POS categories other than Nouns and Verbs. SenSim has a better correlation with gold standard data than other metrics. In fact, all variants of SenSim function better than the existing similarity measurement techniques used in this work. Among different variants, Sen-Sim using TD based scoring function performs the best. It has a correlation score of.50 whereas the nearest correlation among the other metrics is by LCH (.36). Moreover, in case of all POS categories barring Adjectives, SenSim metric has a better correlation than the rest of the metrics with gold standard data.

Although not provided in the table, reader should note that the metrics used in this study could not score all the word pairs created for this study. For example, out of 48 word pairs, SenSim could mark only 34 word pairs and among other metrics, the best count is provided by Lesk with 17 word pairs. The correlation scores shown for each metric is with respect to word pairs that had some values for comparison with the gold standard data. Thus in terms of coverage also, SenSim performs better than the metrics used in this study.

9.5.2 Effect of SenSim on synset replacement strategy

Table 9.5 shows classification result using synset replacement strategy based on similarity metrics. The result of the classifier trained on synsets alone, without any replacement, is taken as the baseline.

SenSim(TM) variant obtains the best classification accuracy among the various metrics analyzed. It achieves an accuracy of 90.17%. Even though the improvement is marginal, reader must note that no complex features are used for training this classifier. All variants of SenSim, except for SenSim(SD) achieve a 90% classification accuracy. Compared to the baseline, other WordNet metrics also have better classification accuracies.

Classification using SenSim (TM) metric has the highest positive precision. Same is the case with negative recall, it has a negative recall of 91.58%. The SenSim (SD) variant has the highest positive recall. In general, it can be seen that positive class's performance has improved by using SenSim metric for synset replacement.

Metric Used	**Accuracy (%)**	**PP**	**NP**	**PR**	**NR**
Baseline	89.10	91.50	87.07	85.18	91.24
LESK (Banerjee *et al.*, 2003)	89.36	91.57	87.46	85.68	91.25
LIN (Lin, 1998)	89.27	91.24	87.61	85.85	90.90
LCH (Leacock *et al.*, 1998)	89.64	90.48	88.86	86.47	89.63
SenSim (SD)	89.95	91.39	88.65	87.11	90.93
SenSim (SM)	90.06	92.01	88.38	86.67	91.58
SenSim (TD)	90.11	91.68	88.69	86.97	91.23

Table 9.5: Classification Results of Synset Replacement Experiment using Different Similarity Metrics

PP-Positive Precision (%), NP-Negative Precision(%), PR-Positive Recall (%), NR-Negative Recall (%) news

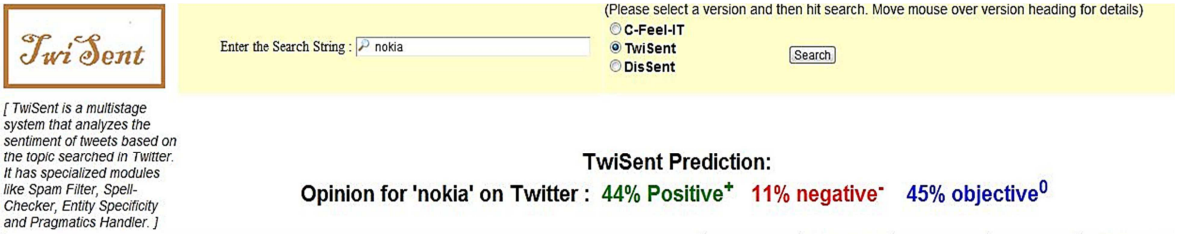
9.6 SUMMARY

We proposed that sentiment content can aid in similarity measurement, which to date have been done on the basis of meaning alone. We verified this hypothesis by taking the correlation between annotators using different annotation strategies. Annotator correlation for the strategy involving sentiment as an additional parameter for similarity measurement was higher than

the one which involved just semantic similarity. Based on this hypothesis, we introduced a new similarity metric, SenSim, which accounts for the sentiment content of the words being compared. An intrinsic evaluation of the metric with human annotated word pairs for similarity showed higher correlations than the popular WordNet based similarity metrics. We also carried out an extrinsic evaluation of SenSim on synset replacement strategy for mitigation of unknown feature problem in supervised classification. Our results suggest that apart from the overall improvement of sentiment classification accuracy, SenSim improves the classification performance of positive class-documents.

Chapter 10

10. WEB INTERFACE



	SentiWordNet	Subjectivity Lexicon	Inquirer	Taboada	Overall Prediction
Nectar site offering comp to win a Nokia Lumia. I'd really like to win! So please DO NOT enter!	+	+	0	+	+
@kaicrawf: I want an iPhone with BBM and the battery life of a Nokia.	+	-	0	+	+
@meghashah0 aggrreeddd !!!! But nokia doesn't offer so many services and the main problem with it is VIRUS.	0	0	+	+	+
This film was shot using the Nokia 808 PureView:	0	0	0	0	0
@EzekielDollar_ Nokia Lumia 800.. . Hope it answers your question :)	+	+	+	+	+
@APNWindowsPhone As soon as they approve it. Their delay, not Nokia's	-	+	0	0	0

Figure 10.1: Twitter Systems Web Interface

The demonstration url[35] [36] [37] points to a web interface which has a field to enter the search string and a radio button to select a system, *C-Feel-It, TwiSent* or *DisSent*. The C-Feel-IT system has been developed by Joshi *et al.* (2011). *TwiSent* refers to the Multi-Stage System for Analyzing Sentiment in Twitter (*Chapter 6*) and *DisSent* refers to the Discourse-based System for Analyzing Sentiment in Twitter (*Chapter 4*). On clicking the search button, the results page shows a snapshot of the tweets retrieved in first column, the individual predictions by sentiment lexicons in columns 2-4 and the final column shows the overall prediction for a given tweet. The page also shows the overall percentage *positive, negative* or *objective* opinion about the search string.

The results page for TwiSent shows much less spams in the form of promotional tweets, bot-generated tweets etc. This can be verified by searching the input string in the Twiiter API and viewing the retrieved tweets, many of which are spams. The effect of entity-specificity can be viewed in some tweets, where the overall predicted tweet sentiment is with respect to the input string. The effect of emoticons and pragmatics, arising out of capitalization and alphabet elongation, is reflected in the overall predicted sentiment of certain tweets. The effect of Spell-Checker can be observed in the tweets which have spelling variations in key sentiment words, which are generally absent in the sentiment lexicons used.

[35] http://www.clia.iitb.ac.in/TwitterSystemsCFILT/index.jsp (can be accessed only from inside IIT Bombay)
[36] http://10.144.22.120:8080/TwitterSystemsCFILT/index.jsp (can be accessed only from inside IIT Bombay)
[37] http://www.clia.iitb.ac.in/TwiSent/ (can be accessed from outside IIT Bombay)

Chapter 11

11. CONCLUSIONS AND FUTURE WORK

11.1 CONCLUSIONS

In this work, we have shown that sentiment analysis is more than just a bag-of-words classification problem. We have analyzed sentiment analysis from different perspectives, and proposed algorithmic approaches to incorporate *discourse coherency*, *feature specificity*, *social content and informal language form,* and *world knowledge* in a sentiment analysis system for better performance.

We proposed a lightweight method of incorporating *discourse features* in a bag-of-words model. Our system achieved better results for unstructured and noisy text in *Twitter* over C-Feel-It (Joshi *et al.*, 2011), as well as for structured reviews in the *Travel* domain over Balamurali *et al.* (2011). Most of the bag-of-words models ignore the *discourse markers* as stop words. In this work, we show how the incorporation of discourse markers can improve the performance of a simple bag-of-words model. The proposed approach has the additional attractiveness of being less resource intensive, suitable to be used by the web-based applications for real-time sentiment analysis.

Feature specificity has been incorporated by considering association between words, forming opinion expressions, using *dependency parsing*. We compared our system to the state-of-the-art systems; and showed that it achieves significant accuracy improvement over Hu *et al.* (2004) and Lin *et al.* (2009), performing at par with Lakkaraju *et al.* (2011) despite its *data limitations*. Although dependency parsing had been used in a few works for feature-specific sentiment analysis, it has not been too much effective. This work establishes the usefulness of dependency relations in extracting feature specific expressions of opinion.

The *social media content* and the *informal language form* in Twitter pose a real challenge to sentiment analysis due to the inherent noise and lack of structure in such domains. We proposed a multi-stage system, *TwiSent*, with specialized modules for handling the nuances of micro-blogging genres, in detecting the sentiment orientation of *tweets*. TwiSent achieved a significant accuracy improvement over C-Feel-It (Joshi *et al.*, 2011) on the same dataset. The major contributions of this work can be summarized as:

1. Categorization of spams in Twitter with respect to sentiment analysis

2. Categorization of the spelling variations and abbreviations encountered in Twitter
3. Suggesting a rule-based module for preserving the *Pragmatics* in Tweets

We proposed a novel method of incorporating world knowledge in a sentiment analysis system for classification of movie reviews. The proposed system, *WikiSent,* uses *Wikipedia* to create a movie-specific, extractive summary of a review. The filtering out of irrelevant concepts helps in better classification of reviews. WikiSent achieves a better accuracy over all the semi-supervised and unsupervised systems in the domain, on the same dataset. This has been the first work, to the best of our knowledge, to attempt topic-specific, extractive summarization of a review using Wikipedia. We also perform a general analysis of the movie domain using WikiSent in terms of the movie genres, the movie release year and the polarity of the reviews to suggest the general trends existing in this domain. Although more than a hundred works have been done on the dataset used (Pang *et al.*, 2002), our work is the first one to analyze it from different perspectives.

YouCat incorporates *world knowledge* from Wikipedia, as in *WikiSent*, and *social media content* from the user posts, as in *TwiSent*, for Youtube video genre prediction. The system achieved a better performance over a supervised classifier despite using *no labeled data for training*. Our system is *mostly unsupervised,* unlike the other works in this domain, which are *mostly supervised*. Although there have been isolated works in video genre prediction using Wikipedia and User comments, our work is the first one to bring all these approaches together, and suggest an unsupervised approach to the problem.

In the final work, we suggest a *sentiment-semantic similarity metric* for unknown feature replacement, which boosts the classifier performance in sentiment analysis of reviews. Most of the similarity metrics, in use, depend on the syntax or the semantics of the word pair for comparison. Our work is the first one to introduce sentiment as a third perspective for similarity computation.

11.2 FUTURE WORK

11.2.1 Mid Term Work

The *discourse-specific* bag-of-words model can be further improved upon for structured reviews with the use of *parsing*, for determining the scope of the discourse markers. Since we proposed a lightweight discourse model, the scope of a discourse marker has been heuristically considered to span the text till another marker or till the sentence boundary. A

ranking of the discourse features, based on their positional importance, and a lexicon with a higher coverage may improve the system performance.

The system accuracy in the *feature specific* sentiment analysis of product reviews suffers due to the usage of a generic lexicon, in the final phase for lexicon-based classification. Usage of a domain-dependent lexicon will boost the system performance.

WikiSent suffers from the lack of handling *synonymous concepts*, *anaphora resolution* and *word sense disambiguation.* Using *WordNet* to detect synonymous concepts and a *WSD* module for fine-grained sense distinction of concepts will add to the system performance. An *anaphora resolution tool* is essential for mapping the pronouns in a sentence to the referred concepts.

A heuristically driven minimum edit-distance based *Spell-Checker* was used in *TwiSent*. *Pragmatics* has been handled with a rule-based module. A more sophisticated approach to Spell-Checker, in presence of a parallel corpora, and Pragmatics Handler may add to the system performance.

YouCat needs to be tested on more number of genres to check its capability for fine-grained genre distinction. The accuracy in YouCat suffers due to the inherent noise in the Youtube media arising out of the user comments and incorrect concept expansion due to ambiguous words. A pre-processing filter that allows only relevant user comments about the video and a WSD module will boost the performance of the system.

11.2.2 Long Term Work

We showed that the performance of the traditional sentiment analysis systems can be improved by incorporating these new perspectives. Our work considers these perspectives one at a time, and shows them to be beneficial in boosting the system performance. Bringing all these perspectives in a single canvas and evaluating their collective effectiveness will pose an interesting research problem.

A more in-depth analysis is required in each of the areas addressed in this work. A root cause analysis needs to be done for all the emerging issues in this field. Most important of all, the notions of sentiment, as to what factors constitute or influence the sentiment of a person need to be thoroughly analyzed. In the movie review domain, it is necessary to analyze the correlation between the opinion of the reviewer with different *features* of the movie, like *acting, direction, cinematography, story-writing etc*. This may involve *topic-modeling* and *user-modeling*. This is because the notion of *significant* features in a movie is very subjective and may vary with *age-group, gender, locality etc*.

12. Publications

Sentiment Analysis in Twitter with Lightweight Discourse Analysis, Subhabrata Mukherjee and Pushpak Bhattacharyya, In Proceedings of the 24th International Conference on Computational Linguistics (**COLING 2012**), IIT Bombay, Mumbai, Dec 8 - Dec 15, 2012 (Long Paper)

YouCat : Weakly Supervised Youtube Video Categorization System from Meta Data & User Comments using WordNet & Wikipedia, Subhabrata Mukherjee and Pushpak Bhattacharyya, In Proceedings of the 24th International Conference on Computational Linguistics (**COLING 2012**), IIT Bombay, Mumbai, Dec 8 - Dec 15, 2012 (Long Paper)

TwiSent: A Multi-Stage System for Analyzing Sentiment in Twitter, Subhabrata Mukherjee, Akshat Malu, Balamurali A.R. and Pushpak Bhattacharyya, In Proceedings of The 21st ACM Conference on Information and Knowledge Management (**CIKM 2012**), Hawai, Oct 29 - Nov 2, 2012 (Short Paper)

WikiSent : Weakly Supervised Sentiment Analysis Through Extractive Summarization With Wikipedia, Subhabrata Mukherjee and Pushpak Bhattacharyya, In Proceedings of the European Conference on Machine Learning (**ECML PKDD 2012**), Bristol, U.K., 24-28 Sept, 2012 (Long Paper)

Feature Specific Sentiment Analysis for Product Reviews, Subhabrata Mukherjee and Pushpak Bhattacharyya, In Proceedings of the 13th International Conference on Intelligent Text Processing and Computational Intelligence (**CICLING 2012**), New Delhi, India, March, 2012 (Long Paper)

Leveraging Sentiment to Compute Word Similarity, Balamurali A.R., Subhabrata Mukherjee, Akshat Malu and Pushpak Bhattacharyya, In Proceedings of the 6th International Global Wordnet Conference (**GWC 2011**), Matsue, Japan, Jan, 2012 (Long Paper)

13. REFERENCES

1. Aditya Joshi, Balamurali A.R., Pushpak Bhattacharyya, 2010, A Fall-Back Strategy for Sentiment Analysis in a New Language: A Case Study for Hindi, ICON 2010, Kharagpur, India
2. Alec, G.; Lei, H.; and Richa, B. Twitter sentiment classification using distant supervision. Technical report, Standford University. 2009
3. Alekh Agarwal and Pushpak Bhattacharyya, Sentiment Analysis: A New Approach for Effective Use of Linguistic Knowledge and Exploiting Similarities in a Set of Documents to be Classified, International Conference on Natural Language Processing (ICON 05), IIT Kanpur, India, December, 2005
4. Alistair Kennedy and Diana Inkpen, Sentiment classification of movie and product reviews using contextual valence shifters, Computational Intelligence, 22(2):110–125, 2006
5. Aone, C., Okurowski, M. E., Gorlinsky, J., and Larsen, B., A trainable summarizer with knowledge acquired from robust nlp techniques, In Mani, I. and Maybury, M. T., editors, Advances in Automatic Text Summarization, pages 71-80, 1999
6. Asher, Nicholas and Benamara, Farah and Mathieu, Yvette Yannick. Distilling opinion in discourse: A preliminary study, In Proceedings of Computational Linguistics (CoLing), 2008
7. B. Wellner, J. Pustejovski, A. Havasi, A. Rumshisky, and R. Suair. 2006, Classification of discourse coherence relations: An exploratory study using multiple knowledge sources, In Proc. of SIGDIAL, 2006
8. Balaji Polepalli P. Ramesh, Hong Yu, Identifying discourse connectives in biomedical text, In Annual Symposium proceedings, AMIA Symposium, Vol. 2010,pp. 657-661, 2010
9. Balamurali A.R., Aditya Joshi, Pushpak Bhattacharyya, Robust Sense Based Sentiment Classification, ACL WASSA 2011, Portland, USA, 2011
10. Balamurali A.R., Aditya Joshi, Pushpak Bhattacharyya, Harnessing WordNet Senses for Supervised Sentiment Classification, In Proceedings of EMNLP, Edinburgh, 2011
11. Banerjee, S. & Pedersen, T., An adapted lesk algorithm for word sense disambiguation using wordnet, in 'Proc. of CICLing'02', London, UK, pp. 136–145, 2002
12. Banerjee, S. & Pedersen, T., Extended gloss overlaps as a measure of semantic relatedness, in 'In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence', pp. 805–810, 2003

13. Barbosa, L., and Feng, J., Robust sentiment detection on twitter from biased and noisy data, In 23rd International Conference on Computational Linguistics: Posters, 36–44, 2010
14. Barzilay, R. and Elhadad, M. (1997), Using lexical chains for text summarization, In Proceedings ISTS'97, 1997
15. Bermingham, A., and Smeaton, A, Classifying sentiment in microblogs: is brevity an advantage ACM 1833–1836, 2010
16. Liu, Bing, Sentiment Analysis and Opinion Mining, 5th Text Analytics Summit, Boston, June 1-2, 2009
17. Pang, Bo and Lee, Lillian, Opinion Mining and Sentiment Analysis, Foundations and Trends in Information Retrieval Volume 2 Issue 1-2, 2008
18. Borth, Damian and Hees, J\"{o}rn and Koch, Markus and Ulges, Adrian and Schulze, Christian and Breuel, Thomas and Paredes, Roberto, TubeFiler – an Automatic Web Video Categorizer, Proceedings of the 17th ACM international conference on Multimedia, MM '09, 2009
19. Boucher, Jerry D. and Charles E. Osgood, The Pollyanna hypothesis, Journal of Verbal Learning and Verbal Behaviour, 8:1–8, 1969
20. Brian Eriksson, SENTIMENT CLASSIFICATION OF MOVIE REVIEWS USING LINGUISTIC PARSING", Processing, Citeseer, Pages: 4-9, Volume 838, 2006
21. C. Whitelaw, N. Garg, and S. Argamon, Using appraisal groups for sentiment analysis, In Proceedings of the ACM SIGIR Conference on Information and Knowledge Management (CIKM), pp. 625–631, ACM, 2005
22. Chen Mosha, Combining Dependency Parsing with Shallow Semantic Analysis for Chinese Opinion-Element Relation Identification, IEEE, pp. 299-305, 2010
23. Christof Müller and Iryna Gurevych, Using Wikipedia and Wiktionary in domain-specific information retrieval, Proceeding CLEF' 08, Springer-Verlag Berlin, Heidelberg, 2009
24. Conroy, J. M. and O'leary, D. P., Text summarization via hidden markov models, In Proceedings of SIGIR '01, pages 406-407, 2001
25. Cui, Bin and Zhang, Ce and Cong, Gao, Content-enriched classifier for web video classification, Proceedings of the 33rd international ACM SIGIR, 2010
26. Daniel M. Bikel, Jeffrey Sorensen, If We Want Your Opinion, International Conference on Semantic Computing (ICSC 2007), 2007

27. Das, D. and Martins, A. F. T. A Survey on Automatic Text Summarization, Literature Survey for the Language and Statistics II course at CMU, Pittsburg 2007
28. David N. Milne and Ian H. Witten and David M. Nichols, A knowledge-based search engine powered by wikipedia, Proceedings of the Sixteenth ACM conference on Conference on information and knowledge management, ACM New York, NY, USA 2007
29. Dey, Lipika and Haque, Sk. Opinion Mining from Noisy Text Data, International Journal on Document Analysis and Recognition 12(3). pp 205-226, 2009
30. Edmundson, H. P., New methods in automatic extracting, Journal of the ACM, 16(2): 264-285, 1969
31. Ekenel, Hazim Kemal and Semela, Tomas and Stiefelhagen, Rainer, Content-based video genre classification using multiple cues, Proceedings of the 3rd international workshop on Automated information extraction in media production, AIEMPro '10, 2010
32. Elwell, Robert and Baldridge, Jason, Discourse Connective Argument Identification with Connective Specific Rankers. In Proceedings of IEEE International Conference on Semantic Computing, 2008
33. Esuli A, Sebastiani F., SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining, In Proceedings from International Conference on Language Resources and Evaluation (LREC), Genoa, 2006
34. Esuli, A. & Sebastiani, F., SentiWordNet: A publicly available lexical resource for opinion mining, In Proceedings of LREC-06, Genova, IT, pp. 417–422, 2006
35. F.Wolf and E. Gibson, Representing discourse coherence: A corpus-based study, Computational Linguistics, 31(2), pp. 249–287, 2005
36. Farah Benamara, Carmine Cesarano, Antonio Picariello, VS Subrahmanian et al; Sentiment Analysis: Adjectives and Adverbs are better than Adjectives Alone, In ICWSM ’2007 Boulder, CO USA, 2007
37. Fei Wu, Daniel S. Weld, Automatically Refining the Wikipedia Infobox Ontology, WWW 2008
38. Filippova, Katja and Hall, Keith B., Improved Video Categorization from Text Metadata and User Comments, Proceedings of the 34th international ACM SIGIR, pp. 835-842, 2011
39. Gabrilovich, Evgeniy and Markovitch, Shaul, Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge,

Proceedings of the 21st national conference on Artificial intelligence - Volume 2, AAAI 2006

40. Giuliano Armano and Alessandro Giuliani and Eloisa Vargiu, Experimenting Text Summarization Techniques for Contextual Advertising, Proceedings of the 2nd Italian Information Retrieval (IIR) Workshop, Milan, Italy, 2011
41. Gonz\'{a}lez-Ib\'{a}\~{n}ez, Roberto and Muresan, Smaranda and Wacholder, Nina. Identifying sarcasm in Twitter: a closer look, In Proc. of ACL: short paper, 2011
42. Grishman, R., Adaptive information extraction and sublanguage analysis, In Proceedings of the 17th International Joint Conference on Artificial Intelligence, 2001
43. Himabindu Lakkaraju, Chiranjib Bhattacharyya, Indrajit Bhattacharya and Srujana Merugu, Exploiting Coherence for the simultaneous discovery of latent facets and associated sentiments, SIAM International Conference on Data Mining (SDM), April 2011
44. Hirst, G. & St-Onge, D., 'Lexical chains as representation of context for the detection and correction malapropisms, 1997
45. Hobbs, Jerry R., and Michael Agar.The Coherence of Incoherent Discourse, Journal of Language and Social Psychology, vol. 4, nos. 3-4, pp. 213-232, 1985
46. Huang, Chunneng and Fu, Tianjun and Chen, Hsinchun, Text-Based Video Content Classification for Online Video-Sharing Sites, Journal of the American Society for Information Science and Technology Volume 61, Issue 5, pages 891–906, 2010
47. J. Rousu and J. Shawe-Taylor, Efficient computation of gapped substring kernels on large alphabets, J. Mach. Learn. Res., 6:1323–1344, 2005
48. Janyce M. Wiebe, Identifying subjective characters in narrative, Proceeding COLING '90, Proceedings of the 13th conference on Computational linguistics- Volume 2, 1990
49. Janyce M. Wiebe, Point of view and discourse processing, Working Notes of the 1991 AAAI Fall Symposium on Discourse Structure in Natural Language Understanding and Generation, 1991
50. Janyce M. Wiebe, William J. Rapaport, A computational theory of perspective and reference in narrative, Proceedings of the 26th annual meeting on Association for Computational Linguistics, 1988
51. Janyce Wiebe M., Identifying subjective characters in narrative, In Proc. 13th International Conference on Computational Linguistics (COLING-90), pp. 401-408, 1990

52. Janyce Wiebe M., References in narrative text, Noûs (Special issue on Cognitive Science and AI) 25 (4), pp. 457-486, 1991
53. Janyce Wiebe M., Tracking point of view in narrative, Computational Linguistics 20 (2), pp. 233-287, 1994
54. Jaynce M. Wiebe, Tracking Point of View in Narrative, Journal Computational Linguistics, Volume 20, Cambridge U.S.A, 1994
55. Jianbo Shi and Jitendra Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8):888–905, 2000
56. Jiang, J. J. & Conrath, D. W., Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy, in 'International Conference Research on Computational Linguistics (ROCLING X)', p. 9008, 1997
57. Joshi, A.; Balamurali, A. R.; Bhattacharyya, P.; and Mohanty, R, C-feel-it: a sentiment analyzer for microblogs, In Proceedings of ACL: Systems Demonstrations, HLT '11, 127–132, 2011
58. Jurafsky, Daniel, and James H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics, Prentice-Hall, 2000
59. K. Dave, S. Lawrence, and D. M. Pennock, Mining the peanut gallery: Opinion extraction and semantic classification of product reviews, In Proceedings of WWW, pp. 519–528, 2003
60. Kimberly Voll and Maite Taboada, Not all words are created equal: Extracting semantic orientation as a function of adjective relevance, In Proceedings of the 20th Australian Joint Conference on Artificial Intelligence, pages 337–346, 2007
61. Leacock, C. & Chodorow, M., Combining local context with wordnet similarity for word sense identification, in WordNet: A Lexical Reference System and its Application, 1998
62. Leacock, C., Miller, G. A. & Chodorow, M., Using corpus statistics and wordnet relations for sense identification, Comput. Linguist. 24, 147–165, 1998
63. Li, Tao and Zhang, Yi and Sindhwani, Vikas, A nonnegative matrix tri-factorization approach to sentiment classification with lexical prior knowledge, In Proceedings of (ACL-IJCNLP), pages 244–252, 2009
64. Lin, C.-Y., Training a selection function for extraction, In Proceedings of CIKM '99, pages 55-62, 1999

65. Lin, Chenghua and He, Yulan and Everson, Richard, A comparative study of Bayesian models for unsupervised sentiment detection, CoNLL, 2010
66. Lin, D., An information-theoretic definition of similarity, in ‘In Proceedings of ICML ’98’, Morgan Kaufmann, pp. 296–304, 1998
67. Luhn, H. P., The automatic creation of literature abstracts, IBM Journal of Research Development. 2(2):159-165, 1958
68. M. Potthast and S. Becker, Opinion Summarization of Web Comments, Proceedings of the 32nd European Conference on Information Retrieval, ECIR, 2010
69. Macdonald, Craig and Ounis, Iadh, Expertise drift and query expansion in expert search, Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, CIKM '07, 2007
70. Mann, William C. and Sandra A. Thompson, Rhetorical Structure Theory: Toward a functional theory of text organization, Text, 8 (3), 243-281, 1988
71. Manning, Christopher D. and Raghavan, Prabhakar and Schtze, Hinrich, Introduction to Information Retrieval. Cambridge University Press, 2008
72. Marcu, D., Improving summarization through rhetorical parsing tuning, In Proceedings of The Sixth Workshop on Very Large Corpora, pages 206-215, pages 206-215, Montreal, Canada, 1998
73. Marcu, Daniel, The Theory and Practice of Discourse Parsing and Summarization, MIT Press, Cambridge, MA, 2000
74. Marcu, Daniel, The Theory and Practice of Discourse Parsing and Summarization, MIT Press, Cambridge, MA., 2000
75. McCarthy, Diana and Koeling, Rob and Weeds, Julie and Carroll, John, Finding Predominant Word Senses in Untagged Text, Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), 2004
76. McCarthy, Diana and Koeling, Rob and Weeds, Julie and Carroll, John, Finding Predominant Word Senses in Untagged Text, Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), 2004
77. Minqing Hu and Bing Liu, Mining and summarizing customer reviews, In Proc. of ACM SIGKDD, 2004
78. Mitesh Khapra, Sapan Shah, Piyush Kedia, and Pushpak Bhattacharyya, Domain-specific word sense disambiguation combining corpus basedand wordnet based parameters, In Proc. of GWC’10, Mumbai, India, 2010

79. Ng, Vincent and Dasgupta, Sajib and Arifin, S. M. Niaz, Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. Proceedings of the COLING/ACL on Main conference poster sessions, 2006
80. P. Turney, Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews, In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02), 2002
81. Pak, Alexander and Paroubek, Patrick, Twitter as a Corpus for Sentiment Analysis and Opinion Mining, LREC, 2010
82. Pang, Bo and Lee, Lillian and Vaithyanathan, Shivakumar, Thumbs up? Sentiment Classification using Machine Learning Techniques, Proceedings of EMNLP 2002
83. Pang, Bo and Lee, Lillian and Vaithyanathan, Shivakumar, Thumbs up?: sentiment classification using machine learning techniques, In Proceedings of the ACL-02 conference on Empirical Methods in Natural Language, 2002
84. Pang, Bo and Lee, Lillian, A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts, Proceedings of the ACL, 2004
85. Patwardhan, S., Incorporating dictionary and corpus information into a context vector measure of semantic relatedness, Master's thesis, University of Minnesota, Duluth, 2003
86. Pedersen, T., Patwardhan, S. & Michelizzi, J., Wordnet::similarity: measuring the relatedness of concepts, in 'Demonstration Papers at HLT-NAACL'04', pp. 38–41, 2004
87. Pitler, Emily and Louis, Annie and Nenkova, Ani. Automatic Sense Prediction for Implicit Discourse Relations in Text, In Proc. of ACL and IJCNLP, 2009
88. Polanyi, Livia and Zaenen, Annie, Contextual valence shifters, In Exploring Attitude and Affect in Text: Theories and Applications, AAAI Spring Symposium Series, 2004
89. Prelovac, V., Top social media sites. Web, 2010
90. Qi Zhang, Yuanbin Wu, Tao Li, Mitsunori Ogihara, Joseph Johnson, Xuanjing Huang, Mining Product Reviews Based on Shallow Dependency Parsing, Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, 2009
91. R. Soricut and D. Marcu, Sentence level discourse parsing using syntactic and lexical information, In Proc of HLT-NAACL, 2003
92. Rada, R., Mili, H., Bicknell, E. & Blettner, M., Development and application of a metric on semantic nets', IEEE Transactions on Systems Management and Cybernetics 19(1), 17–30, 1989

93. Ramanathan Narayanan, Bing Liu and Alok Choudhary, Sentiment Analysis of Conditional Sentences, In Proceedings of Conference on Empirical Methods in Natural Language Processing, 2009
94. Read, Jonathon, Using emoticons to reduce dependency in machine learning techniques for sentiment classification, In ACL. The Association for Computer Linguistics, 2005
95. Resnik, P., Disambiguating noun groupings with respect toWordnet senses, in D. Yarovsky & K. Church, Proceedings of the Third Workshop on Very Large Corpora, Association for Computational Linguistics, Somerset, New Jersey, pp. 54–68, 1995
96. Resnik, P., Using information content to evaluate semantic similarity in a taxonomy, In 'In Proceedings of the 14th International Joint Conference on Artificial Intelligence', pp. 448–453, 1995
97. Richardson, R., Smeaton, A. F. & Murphy, J., Using wordnet as a knowledge base for measuring semantic similarity between words, Technical report, In Proceedings of AICS Conference, 1994
98. S. Zanetti, L. Zelnik-Manor, and P. Perona, A walk through the web's video clips, In Proc. of CVPR Workshop on Internet Vision, 2008
99. Sajib Dasgupta and Vincent Ng, Topic-wise, Sentiment-wise, or Otherwise? Identifying the Hidden Dimension for Unsupervised Text Classification, EMNLP, 2009
100. Socher, Richard and Pennington, Jeffrey and Huang, Eric H. and Ng, Andrew Y. and Manning, Christopher D., Semi-supervised recursive autoencoders for predicting sentiment distribution, EMNLP, 2011
101. Somasundaran, Swapna, Discourse-level relations for Opinion Analysis, PhD Thesis, University of Pittsburgh, 2010
102. Song, Yicheng and Zhang, Yong-dong and Zhang, Xu and Cao, Juan and Li, Jing-Tao, Google Challenge: Incremental-Learning for Web Video Categorization on Robust Semantic Feature Space, Proceedings of the 17th ACM International conference on Multimedia, 2009
103. Stefano Baccianella and Andrea Esuli and Fabrizio Sebastiani, SENTIWORDNET 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining, LREC, 2010
104. Stone, P.J., Dunphy, D.C., Smith, M.S., Ogilvie, D.M. and associates, The General Inquirer: A Computer Approach to Content Analysis. The MIT Press, 1966

105. T. Mullen and N. Collier, Sentiment analysis using support vector machines with diverse information sources, In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 412–418, 2004
106. Taboada, Maite and Brooke, Julian and Tofiloski, Milan and Voll, Kimberly and Stede, Manfred, Lexicon-based methods for sentiment analysis, Computational Linguistics, 2011
107. Taboada, Maite and Brooke, Julian and Voll, Kimberly, Extracting Sentiment as a Function of Discourse Structure and Topicality, Simon Fraser Univeristy School of Computing Science Technical Report, 2008
108. Tetsuji Nakagawa and Kentaro Inui and Sadao Kurohashi, Dependency tree-based sentiment classification using CRFs with hidden variables, NAACL, 2010
109. Tetsuya Nasukawa, Jeonghee Yi, Sentiment Analysis: Capturing Favorability Using Natural Language Processing, In K-CAP ’03, pages 1-8, Florida, 2003
110. Vipul Pandey and C.V.Krishnakumar Iyer, Sentiment Analysis of Microblogs, CS 229 Project Report, Stanford University
111. Wan, S. & Angryk, R. A., Measuring semantic similarity using wordnet-based context vectors., in ‘SMC’07’, pp. 908–913, 2007
112. Wang H., Zhou G., Topic-driven Multi-Document Summarization, In Proceedings International Conference on Asian Language Processing, 2010
113. Wang, Pu and Domeniconi, Carlotta, Building semantic kernels for text classification using Wikipedia, Proc. of SIGKDD, 2008
114. Webber, Bonnie and Knott, Alistair and Stone, Matthew and Joshi, Aravind, Discourse relations: A structural and presuppositional account using lexicalized tag, In Proceedings of ACL, 1999
115. Wellner, Ben and Pustejovsky, James and Havasi, Catherine and Rumshisky, Anna and Saur\'{\i}, Roser, Classification of discourse coherence relations: an exploratory study using multiple knowledge sources, Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue, 2006
116. Wolf, F., Gibson, E. and Desmet, T., Discourse coherence and pronoun resolution, Language and Cognitive Processes, 19(6), pp. 665–675, 2004
117. Wolf, Florian and Gibson, Edward, Representing discourse coherence: A corpus-based study, Computational Linguistics, 31(2), pp. 249–287, 2005

118. Wu, Xiao and Ngo, Chong-Wah and Zhu, Yi-Ming and Peng, Qiang, Boosting web video categorization with contextual information from social web, World Wide Web Volume 15 Issue 2, 2012
119. Wu, Z. & Palmer, M., Verb semantics and lexical selection, in '32nd. Annual Meeting of the Association for Computational Linguistics', New Mexico State University, Las Cruces, New Mexico, pp. 133 –138, 1994
120. Yang, Linjun and Liu, Jiemin and Yang, Xiaokang and Hua, Xian-Sheng, Multi-Modality Web Video Categorization, Proceeding MIR '07 Proceedings of the international workshop on Workshop on multimedia information retrieval, 2007
121. Yew, Jude and Shamma, David A. and Churchill, Elizabeth F., Knowing funny: genre perception and categorization in social video sharing, In Proceedings of CHI'2011, pp.297-306, 2011
122. Yu, Hong and Vasileios, Hatzivassiloglou, Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences, In EMNLP, 2003
123. Yuanbin Wu, Qi Zhang, Xuanjing Huang, Lide Wu, Phrase Dependency Parsing for Opinion Mining, Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, 2009
124. Zhang, John R. and Song, Yang and Leung, Thomas, Improving Video Classification via YouTube Video Co-Watch Data, ACM Workshop on Social and Behavioural Network Media Access at ACM MM 2011
125. Zhang, Ruofei and Sarukkai, Ramesh and Chow, Jyh-Herng and Dai, Wei and Zhang, Zhongfei, Joint Categorization of Queries and Clips for Web-based Video Search, Proceeding MIR '06 Proceedings of the 8th ACM international workshop on Multimedia information retrieval, 2006
126. Zheshen Wang, Ming Zhao, Yang Song, Sanjiv Kumar, and Baoxin Li, YouTubeCat: Learning to Categorize Wild Web Videos, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2010
127. Zhong, Z. & Ng, H. T., It makes sense: A wide-coverage word sense disambiguation system for free text. In 'ACL (System Demonstrations)' 10', pp. 78–83, 2010
128. Zirn, C\"{a}cilia and Niepert, Mathias and Stuckenschmidt, Heiner and Strube, Michael, Fine-Grained Sentiment Analysis with Structural Features, In Proc. of IJCNLP. 2011