An Enhanced Feature-based Sentiment Analysis System

GRADUATE PROJECT REPORT

Submitted to the Faculty of
The School of Engineering & Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, TX

in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

by

Srividya Venumbaka
Spring 2013

Committee Members

Dr. Longzhuang Li
Committee Chairperson

Dr. Dulal Kar
Committee Member

Dr. David Thomas
Committee Member
ABSTRACT

Due to the increased usage of internet sources such as websites and forums, sentiment analysis have become a challenging research area in the past decade. There are many feature-based sentiment analysis approaches where features can be the word itself, or its part-of-speech, or some polarity tags. However, the lexical resources they use (positive and negative words list) does not generate accurate opinion extraction results. Another important problem is to solve complex opinion structures (eg. I do ‘not’ think it is a ‘bad’ product) where, just dealing with words separately is not sufficient.

The proposed solution overcomes these limitations and improves the accuracy of opinion extraction. This system accepts two types of input: text files and URLs from a website (Amazon is considered). Preparing data and building processing components are the main stages of work focused here. Preparing data includes building positive words list, negative words list and lists with words that can invert, increase or decrease the opinion and also enhancing these lists using SentiWordNet. The second stage, building processing components, takes a URL as input and determines the product and the comments. An open source tool called Stanford is used for stemming and parts-of-speech (POS) tagging. Also, opinion tags and special tags generated through Transformation-Based Learning (TBL) are used to invert, increase or decrease the opinion. The final opinion is determined by aggregating the opinion weights at word-level, sentence-level and document-level.
# TABLE OF CONTENTS

Abstract ........................................................................................................................................... ii

Table of Contents ................................................................................................................................ iii

List of Figures ..................................................................................................................................... v

List of Tables ....................................................................................................................................... vi

1. Background and Rationale .............................................................................................................. 1
   1.1 Sentiment Analysis ....................................................................................................................... 1
   1.2 Feature-based Sentiment Analysis .................................................................................................. 2
   1.3 Existing Feature-based Sentiment Analysis Systems ................................................................. 3
       1.3.1 Opinion Summarization System based on POS Tagging ................................................. 3
       1.3.2 Review Classification using SentiWordNet Lexicon ......................................................... 4
       1.3.3 Semantic Orientation CALculator (SO-CAL) .................................................................... 6
       1.3.4 Sentence-level Subjectivity using SVM .............................................................................. 7
       1.3.5 OPINE – A High Precision Opinion Extraction System .................................................... 7

2. Narrative .......................................................................................................................................... 9
   2.1 Problem Statement ...................................................................................................................... 9
   2.2 Motivation ................................................................................................................................. 9
   2.3 Project Objective ...................................................................................................................... 9
   2.4 Functionalities of the Project ................................................................................................... 10

3. Proposed System Design ............................................................................................................... 11
   3.1 Steps in the Project Development ............................................................................................ 12
       3.1.1 Stemming .......................................................................................................................... 12
       3.1.2 POS Tagging .................................................................................................................... 13
3.1.3 Word-level Opinion Tagging ................................................................. 15
3.1.4 Enriching Tags ..................................................................................... 18
3.1.5 Sentence-level Opinion Mining ............................................................ 22
3.1.6 Document-level Opinion Mining ......................................................... 23
3.2 Environment ............................................................................................ 24
  3.2.1 Java Swing ......................................................................................... 24
  3.2.2 Jsoup Parser ..................................................................................... 25
  3.2.3 NetBeans IDE ................................................................................... 25
3.3 Software Modules ..................................................................................... 26
4. Implementation and Results ....................................................................... 28
  4.1 Reviews Entered by User as Input .......................................................... 28
  4.2 Text File with Customer Reviews as Input ............................................ 31
  4.3 Amazon Webpage (URL) with Customer Reviews as Input ................. 35
5. Testing and Evaluation ............................................................................... 39
6. Conclusion and Future Work ..................................................................... 42
Bibliography and References ......................................................................... 43
LIST OF FIGURES

Figure 1.1. The Opinion Summarization System ........................................4
Figure 1.2. Sentiment Classification Phases ..............................................5
Figure 3.1. Flow Diagram of the Proposed System ....................................12
Figure 3.2. Few Entries of SentiWordNet ..............................................15
Figure 3.3. Entries of ‘good’ as Adjective in SentiWordNet .....................16
Figure 3.4. Entries of ‘good’ as Noun and Adverb in SentiWordNet ..........17
Figure 3.5. Document-level Output View .............................................24
Figure 3.6. User-Interface showing Input Handler..................................27
Figure 4.1. Reviews Entered by the user as Input ..................................28
Figure 4.2. Output for the Reviews Entered by the User .........................29
Figure 4.3. Backend Processing of the Reviews Entered by the User .........30
Figure 4.4. User Selecting Text File as Input .....................................31
Figure 4.5. Text File given as Input .....................................................32
Figure 4.6. Output for the Text File Input ...........................................33
Figure 4.7. Backend Processing of the Text File Input ............................34
Figure 4.8. Amazon Webpage Considered as Input .................................35
Figure 4.9. Amazon URL Entered as Input .........................................36
Figure 4.10. Output for the URL Input Entered .....................................37
Figure 4.11. Backend Processing of the URL Input .................................38
Figure 5.1. System Output of the Test Cases Considered .........................40
LIST OF TABLES

Table 3.1. Treebank Tagset (a) .....................................................................................14
Table 3.2. Treebank Tagset (b) .....................................................................................15
Table 5.1. Evaluation Report ..........................................................................................39
1. BACKGROUND AND RATIONALE

1.1 Sentiment Analysis

Text mining is a variation of data mining. Data mining deals with the extraction of interesting patterns from large structured databases, whereas text mining deals with extraction of interesting patterns from natural language text. Databases can be processed automatically by programs, but, text can be read only by people. There are no programs that can read text. However, Natural Language Processing (NLP) or Computational Linguistics can be used for text analysis. This field is used to extract phrases or words from text and summarize its contents.

Facts and opinions are the two main categories of textual information. Facts are objective expressions about events, entities and their properties. Opinions are subjective expressions describing people’s sentiments or feelings about events, entities and their properties. Mining opinion expressions is mainly focused in this project. Opinion mining also called as sentiment analysis is used to extract sentiment or customers’ opinion from the text of their reviews regarding a product or service. Business organizations always want to find the consumer or public opinions about their services and products [1]. Customers also want to know the existing users’ opinions before purchasing a product or using a service. Internet is the most widely used medium for product publicity, exchanging information and opinions. With the rapid growth of social media such as forums and blogs on the Web, business organizations and individuals are using public opinions posted in these media for their decision making and product improvements.
Since each website or forum contains a huge volume of product reviews (opinionated text), normal users and organizations will have difficulty summarizing opinions.

Machine learning approaches and approaches based on NLP and lexical resources are the two types of techniques used to classify product reviews. Machine learning approaches include Support Vector Machine (SVM) and Naïve Bayesian classification [2], [3]. SVM is a supervised learning model used mainly to analyze the data and recognize data patterns that can be utilized for classification and regression analysis. Naïve Bayesian Classification is based on Naïve-Bayes theorem. It uses the concepts of maximum likelihood and bayesian probability. Approaches based on NLP and lexical resources, also called as feature-based sentiment analysis approaches, mainly use parts of speech information and WordNet [4], [5], [6]. There are few hybrid approaches that combine machine learning and lexical resources [7], [8].

1.2 Feature-based Sentiment Analysis

Features can be just the words, or their parts of speech (POS) or some polarity tags. Feature-based sentiment analysis models first discover the objects in a sentence on which, opinions have been expressed and then determine if the opinions are positive, negative or neutral [9]. An object can be a topic, event, product, service, etc. In sentiment analysis, features can be explored at word-level [10], phrase-level [11], sentence-level [12], paragraph-level [13] and document-level [14]. Also, features can be categorized into syntactic, semantic, link-based and statistic features [15]. Syntactic features include words, POS tags, punctuation marks and phrase patterns. Semantic features include polarity tags, semantic orientation and appraisal groups. Link-based features include Web
links, document citations and send/reply patterns. Statistic features include sentence length, word count and delimiter count.

1.3 Existing Feature-based Sentiment Analysis Systems

Different existing feature-based sentiment analysis systems are discussed in this section.

1.3.1 Opinion Summarization System based on POS Tagging

This system mainly concentrates on specific features (parts) of a product that customers expressed opinions on [16]. It summarizes opinions in two steps: feature extraction and opinion direction identification. Figure 1.1 shows the Opinion Summarization System. POS tagging is parts of speech tagging which is implemented using NLProcessor linguistic parser (NLProcessor 2000). It parses each sentence and tags each word with its part of speech (noun/verb/adjective etc). Frequent feature generation is done by using association rule mining. Feature pruning removes all redundant features. Opinion words extraction is based on the observation that “opinions appear close to features”. Semantic orientation of each word in the opinion words list is identified by using WordNet and bootstrapping technique. Finally, opinion orientation of each sentence is decided based on the dominant orientation of opinion words in the sentence.
Disadvantage

This system does not handle negations and thus cannot solve complex opinion structures.

1.3.2 Review Classification using SentiWordNet Lexicon

SentiWordNet is a lexical resource which assigns the three numerical sentiment scores, positivity, negativity and neutrality/objectivity to each synset of WordNet [17]. The range of each of the three scores is 0.0 – 1.0 and their sum for each synset is 1.0. This approach consists of two phases: SentiwordNet Interpretation phase and Sentiment Polarity calculation Phase [18]. In the first phase, positive and negative scores are assigned to the words. In the second phase, magnitude of positive and negative scores is considered to classify the reviews as positive and negative.
Figure 1.2. Sentiment Classification Phases

Figure 1.2 shows the sentiment classification phases along with the preparation phase. The positive and negative scores for each word in SentiWordNet lexicon, are calculated by finding the average for its entries according to parts of speech category: noun, adjective, verb and adverb. Magnitude value of positive and negative scores is considered by using term score summation method, average on sentence and average on review methods in the Sentiment Polarity Calculation phase. Tokenization process, sentence splitting process and speech tagging process are parts of the preparation phase. Tokenization process splits the text into simple tokens such as punctuation and numbers.
Sentence splitting process splits the text into sentences based on full stop marks. Speech tagging process tags each word with its part of speech.

Disadvantages

This approach uses a threshold to ignore words with high neutral scores. This decreases the accuracy of review classification. Also, negation expressions to exchange polarity of the sentence are not considered in this approach.

1.3.3 Sentiment Orientation CALculator (SO-CAL)

Semantic Orientation (SO) measures the opinion and subjectivity in text. SO-CAL considers nouns, verbs, adverbs and adjectives [19]. Previous versions of SO-CAL considered adjectives only. The two main assumptions of SO-CAL are:

- Individual words have semantic orientation (prior polarity) which is independent of context.
- Semantic orientation can be represented as a numerical value.

SO-CAL uses dictionaries of words annotated with their evaluative factors, polarity and strength. It incorporates the two main categories of intensifiers: amplifiers, which increase the semantic intensity and downtoners, which decrease the semantic intensity of the nearest lexical item. It also handles negation to revert the polarity of the lexical item next to the negator.

Disadvantage

This approach is good only at word-level and sentence-level sentiment classification. It is not accurate in cases of paragraph-level and document-level sentiment classification.
1.3.4 Sentence-level Subjectivity using SVM

This approach mainly focused on sentiment analysis and subjectivity in newspapers [12]. It involves the two subtasks: opinionated subtask and polarity subtask. Opinionated subtask concentrates on classifying sentences with opinions and without opinions. A trained model is used in this subtask, to label each token as an opinion expression. Polarity subtask assigns polarities (positive, negative or neutral) to each sentence or subsentence with a distinct opinion. The trained models label each token with a certain polarity. In both the subtasks, the token-level labeling predictions are propagated to the sentence-level classification results using some heuristics. This approach makes use of TreeTagger POS tagger, the Stanford Dependency Parser and SVM.

Disadvantage

This approach is good only at sentence-level sentiment classification.

1.3.5 OPINE – A High Precision Opinion Extraction System

This is an unsupervised opinion extraction system which mines opinions of important product features [20]. OPINE provides solution to each of the following subtasks:

- Identifying the product features.
- Identifying the opinions according to product features.
- Determining the opinion polarity.
- Ranking opinions based on their strength.
The main goal of OPINE is to find the set of (feature, opinions) tuples. It determines Semantic Orientation (SO) of a word ‘w’ with respect to a feature ‘f’ and associated sentence ‘s’. This is done in three steps:

- Opine determines a SO label ({positive, negative, neutral}) for each word w, when a set of reviews are given.
- Opine determines a SO label for each (w, f) pair, when a set of reviews are given.
- Opine determines a SO label for each (w, f, s) tuple, when a set of SO labels for (w, f) pairs are given.

Disadvantage

This system concentrates only on extracting opinions of specific features of a product rather than opinion of the product as a whole.
2. NARRATIVE

2.1 Problem Statement

Most of the existing feature-based sentiment analysis systems are good only at word-level and sentence-level sentiment analysis. The lexical resources they use, i.e., the list of opinion words and phrases, do not cover all the expressions that imply or convey opinions. This generates opinion extraction results which may not be always accurate (poor accuracy). Another important problem is to solve complex opinion structures (eg. I do ‘not’ think it is a ‘bad’ product) where, just dealing with the words is not sufficient.

2.2 Motivation

Business organizations always want to find the consumer or public opinions about their services and products. Customers also want to know the existing users’ opinions before purchasing a product or using a service. “What other people think” has always been important for most of the people during their decision making process. 73% - 87% of internet users report that online reviews had a great influence on their purchase [21]. Thus, the need for an efficient feature-based sentiment analysis system is increasing with the growing availability of opinion-rich online resources such as review sites and personal blogs.

2.3 Project Objective

The main objective of the proposed system is to increase (improve) the accuracy of opinion extraction results at document-level and to solve complex opinion structures.
This is done by enhancing the lists of positive, negative, neutral words using SentiWordNet and applying special (enriched) tags to handle intensification as well as negation. Stemming and POS tagging are also used.

2.4 Functionalities of the Project

Sentiment analysis is part of mission of good number of small and large companies. It is very useful in business intelligence to improve sales rate and in government intelligence applications. It is also widely used in politics and eRulemaking. In politics, opinion mining deals with what voters are thinking and what public figures support or oppose. Opinion mining in eRulemaking deals with public opinions regarding government-regulation proposals or pending policies.
3. PROPOSED SYSTEM DESIGN

The proposed feature-based sentiment analysis system consists of two main stages: preparing data and building processing components. In the first stage, positive words list, negative words list and lists with words that can invert, increase or decrease the opinion are built and these lists are enhanced using SentiWordNet. The second stage, building processing components, takes a URL (from Amazon website) as input and determines the product and the comments. An open source tool called Stanford is used for stemming and parts-of-speech tagging. Stemming reduces/maps all derived and related words to their root/stem. Opinion tags and special tags to invert (handling negations), increase or decrease (handling intensification) the opinion are used. If ‘Pos’ is a positive opinion word/phrase and ‘Neg’ is a negative opinion word/phrase, these special tags can be represented as:

- Negation Neg → Positive
- Negation Pos → Negative
- Increased Neg → Negative
- Increased Pos → Positive
- Decreased Neg → Positive
- Decreased Pos → Negative

These special tags are generated using Transformation-Based Learning (TBL), where, a tag is initially assigned to each word and later changed using a set of defined rules [22], [23]. This process also identifies nouns and adjectives. The final opinion is determined by aggregating the opinion weights at word-level, sentence-level and document-level.
Thus, the proposed system is overall a system which first defines and then summarizes the opinions. This system accepts two types of input: text files and URLs from Amazon website. In this system, features are explored at word-level, sentence-level and document-level. The flow diagram of the proposed system is shown in Figure 3.1. All the steps in the project development are explained in detail in the following subsection.

![Flow Diagram of the Proposed System](image)

**Figure 3.1. Flow Diagram of the Proposed System**

### 3.1 Steps in the Project Development

#### 3.1.1 Stemming

Once the input is given, either through a text document or an Amazon URL, Maxent Tagger, a part of Stanford NLP Tagger, is used to split the sentences into words.
Stemming is the process of reducing a word (sometimes derived word) to its root or stem form. This results in all the related words mapping to same stem. This process reduces the number of distinct terms used to represent documents, which in turn saves storage space and the time taken to process them. Each word should be tagged with its stem. This is done by using Stanford tool [25]. Every word in all the sentences from the given input is tagged with its stem by the end of this step. Every word in all the sentences from the given input is tagged with its stem by the end of this step. Consider the following input with three sentences for which stemming and all the further steps are explained clearly.

*Input*: This phone is good and easy to use. The maps are very helpful in searching new places. But it is difficult to carry.

- *Sentence 1*: This/[this] phone/[phone] is/[be] good/[good] and/[and] easy/[easy] to/[to] use/[use].
- *Sentence 3*: But/[but] it/[it] is/[be] difficult/[difficult] to/[to] carry/[carry].

### 3.1.2 POS Tagging

POS tagging is the process of assigning each word, a particular parts of speech such as noun, verb, adjective etc based on its definition and the context in which it is used. The words tagged with their stems by the end of first step, stemming, should also be tagged with their corresponding parts-of speech. This is done by using Stanford POS tagger [26]. Consider the same input used in previous step. Following is the output of this step.
- **Sentence 1:** This/[DT] phone/[NN] is/[VBZ] good/[JJ] and/[CC] easy/[JJ] to/[TO] use/[VB].

- **Sentence 2:** The/[DT] maps/[NNS] are/[VBP] very/[RB] helpful/[JJ] in/[IN] searching/[VBG] new/[JJ] places/[NNS].

- **Sentence 3:** But/[CC] it/[PRP] is/[VBZ] difficult/[JJ] to/[TO] carry/[VB].

DT is the determiner, VBZ is a verb (singular present), JJ is adjective, CC is a coordinating conjunction, VB is the verb (base form), NN is a singular noun etc. Some of the most important POS tags are shown in the following tables Table 3.1 and Table 3.2. These are considered from Penn Treebank Tagset [27].

**Table 3.1. Treebank Tagset (a)**

<table>
<thead>
<tr>
<th>POS Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>coordinating conjunction</td>
<td>and</td>
</tr>
<tr>
<td>CD</td>
<td>cardinal number</td>
<td>1, two</td>
</tr>
<tr>
<td>DT</td>
<td>determiner</td>
<td>the</td>
</tr>
<tr>
<td>EX</td>
<td>existential there</td>
<td>there is</td>
</tr>
<tr>
<td>FW</td>
<td>foreign word</td>
<td>elsewhere</td>
</tr>
<tr>
<td>IN</td>
<td>preposition subordinating conjunction</td>
<td>of, of the</td>
</tr>
<tr>
<td>JJ</td>
<td>adjective</td>
<td>green</td>
</tr>
<tr>
<td>JJR</td>
<td>adjective, comparative</td>
<td>greater</td>
</tr>
<tr>
<td>JJT</td>
<td>adjective, superlative</td>
<td>greatest</td>
</tr>
<tr>
<td>LS</td>
<td>list marker</td>
<td>)</td>
</tr>
<tr>
<td>MD</td>
<td>modal</td>
<td>could, will</td>
</tr>
<tr>
<td>NN</td>
<td>noun, singular or mass</td>
<td>table</td>
</tr>
<tr>
<td>NNS</td>
<td>noun plural</td>
<td>tables</td>
</tr>
<tr>
<td>NNP</td>
<td>proper noun, singular</td>
<td>John</td>
</tr>
<tr>
<td>NNPS</td>
<td>proper noun, plural</td>
<td>Vikings</td>
</tr>
<tr>
<td>PDT</td>
<td>predeterminer</td>
<td>all, the boys</td>
</tr>
<tr>
<td>POS</td>
<td>possessive ending</td>
<td>'s</td>
</tr>
</tbody>
</table>
Table 3.2. Treebank Tagset (b)

<table>
<thead>
<tr>
<th>Tag</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PP</td>
<td>personal pronoun</td>
</tr>
<tr>
<td>PPS</td>
<td>possessive pronoun</td>
</tr>
<tr>
<td>RB</td>
<td>adverb</td>
</tr>
<tr>
<td>RBV</td>
<td>adverb, comparative</td>
</tr>
<tr>
<td>RBZ</td>
<td>adverb, comparative</td>
</tr>
<tr>
<td>RF</td>
<td>particle has got</td>
</tr>
<tr>
<td>V</td>
<td>verb</td>
</tr>
<tr>
<td>VBG</td>
<td>verb, past tense</td>
</tr>
<tr>
<td>VBN</td>
<td>verb, past participle</td>
</tr>
<tr>
<td>VBP</td>
<td>verb, non-past participle</td>
</tr>
<tr>
<td>VBP</td>
<td>verb, non-past participle</td>
</tr>
<tr>
<td>VBD</td>
<td>verb, base form</td>
</tr>
<tr>
<td>VBD</td>
<td>verb, base form</td>
</tr>
<tr>
<td>VBD</td>
<td>verb, base form</td>
</tr>
<tr>
<td>VBD</td>
<td>verb, base form</td>
</tr>
<tr>
<td>VBD</td>
<td>verb, base form</td>
</tr>
<tr>
<td>N</td>
<td>noun, nounal part</td>
</tr>
<tr>
<td>DT</td>
<td>determiner</td>
</tr>
<tr>
<td>WP</td>
<td>preposition</td>
</tr>
<tr>
<td>WP</td>
<td>preposition</td>
</tr>
<tr>
<td>WP</td>
<td>preposition</td>
</tr>
</tbody>
</table>

Each word in all sentences from the given input is tagged with its stem and POS by the end of this step. These words are used in the next step to tag with opinions.

3.1.3 Word-level Opinion Tagging

SentiWordNet is an open source lexical resource which has positivity and negativity assigned to each synset [28]. It is used to enhance the word lists built initially. There are about 1,18,000 entries in SentiWordNet. Few entries are shown in Figure 3.2.

Figure 3.2. Few Entries of SentiWordNet
Each synset term has positivity, negativity and a sense number, as each term can have different categories such as noun, verb, adjective etc and many distinct meanings in each category. For example, the term ‘good’ can be an adjective, noun or adverb. It has 21 entries as adjective, 4 entries as noun and 2 entries as adverb in SetiWordNet. This is clearly shown in Figure 3.3 and Figure 3.4.

**Figure 3.3. Entries of ‘good’ as Adjective in SentiWordNet**
The words from the previous step are compared with the word lists built and SentiWordNet. Opinion tags are assigned to each word accordingly such as opn_positive, opn_negative, neg, inc or dec. Now, each word is assigned with its stem, POS and opinion. Consider the same input used in previous step. Following is the output after word-level opinion tagging.

- **Sentence 1**: This/[POS_DT][Stm_this] phone/[POS_NN][Stm_phone] is/[POS_VBZ][Stm_be] good/[POS JJ][Stm_good][Opn_positive][pos] and/[POS_CC][Stm_and] easy/[POS JJ][Stm_easy][pos] to/[POS_TO][Stm_to][pos] use/[POS_VB][Stm_use][pos].

- **Sentence 2**: The/[POS_DT][Stm_the] maps/[POS_NNS][Stm_map] are/[POS_VBP][Stm_be] very/[POS_RB][Stm_very][inc] helpful/[POS JJ]
Sentence 3: But it is difficult to carry.

3.1.4 Enriching Tags

This step uses the following rules that we developed. The main purpose of developing these rules is to handle the opinion negation and intensification cases. Also, these rules are very helpful in solving complex opinion structures. This step handles opinion negation (by changing pos to neg or neg to pos) and opinion intensification cases such as marking the words as less positive (lpos), positive (pos), very positive (vpos), less negative (lneg), negative (neg) or very negative (vneg). The words only with opinion tags are pointed to 0. The notations 0, -1 etc in these rules represents the word at current position and the word before it respectively. The terms inc, dec and inv denote words from the list of words that increase, decrease and invert the opinion respectively. Also, ‘ineg’ specifies that a positive word is inverted to negative and ‘ipos’ specifies that a negative word is inverted to positive. + or – before any of the terms defined above denote specify that ‘they exist’ or ‘do not exist’.

- [0, +Opn_positive] => [0, +pos]

This rule states that, all the rules with pos in left hand side of the rules are also applicable to the rules with Opn_postive in left hand side of the rules.

- [0, +Opn_negative] => [0, +neg]
This rule states that, all the rules with neg in left hand side of the rules are also applicable to the rules with Opn_negative in left hand side of the rules.

- \([0, +\text{vpos}]=\>[0, +\text{pos}]\)
  
  This rule states that, all the rules with pos in left hand side of the rules are also applicable to the rules with vpos in left hand side of the rules.

- \([0, +\text{vneg}]=\>[0, +\text{neg}]\)
  
  This rule states that, all the rules with neg in left hand side of the rules are also applicable to the rules with vneg in left hand side of the rules.

- \([0, +\text{Opn\_strong\_positive}]=\>[0, +\text{vpos}]\)
  
  This rule states that, all the rules applicable to vpos are also applicable to the rules with Opn\_strong\_positive in left hand side of the rules.

- \([0, +\text{Opn\_strong\_negative}]=\>[0, +\text{vneg}]\)
  
  This rule states that, all the rules applicable to vneg are also applicable to the rules with Opn\_strong\_negative in left hand side of the rules.

- \([-n, +\text{inc}]\&[0, +\text{pos}]=\>[\-n, .]\&[0, +\text{vpos}]\) for \(n=1, 2, 3,\ldots\)etc
  
  This set of rules state that, if the word at position 0 has a positive word and the word at \(-n\) has a word from the list of words that increase the opinion, we leave the word at \(-n\) as it is and change the opinion of word at 0 from pos to vpos.

  Consider an example for \(n=1\).

  **Sentence:** This is a very[inc] good[pos] product.

  **Sentence after enriching tags:** This is a very[inc] good[vpos] product.

- \([-n, +\text{inc}]\&[0, +\text{neg}]=\>[\-n, .]\&[0, +\text{vneg}]\) for \(n=1, 2, 3,\ldots\)etc
This set of rules state that, if the word at position 0 has a negative word and the word at \(-n\) has a word from the list of words that increase the opinion, we leave the word at \(-n\) as it is and change the opinion of word at 0 from neg to vneg. Consider an example for \(n=1\).

Sentence: This is a very[inc] bad[neg] product.

Sentence after enriching tags: This is a very[inc] bad[vneg] product.

- \([-n, +dec] \& [0, +pos] \Rightarrow [-n, .] \& [0, +lpos]\) for \(n=1, 2, 3, \ldots\) etc

This set of rules state that, if the word at position 0 has a positive word and the word at \(-n\) has a word from the list of words that decrease the opinion, we leave the word at \(-n\) as it is and change the opinion of word at 0 from pos to lpos. Consider an example for \(n=2\).

Sentence: This is somewhat[dec] a good[pos] product.

Sentence after enriching tags: This is somewhat[dec] a good[lpos] product.

- \([-n, +dec] \& [0, +neg] \Rightarrow [-n, .] \& [0, +lneg]\) for \(n=1, 2, 3, \ldots\) etc

This set of rules state that, if the word at position 0 has a negative word and the word at \(-n\) has a word from the list of words that decrease the opinion, we leave the word at \(-n\) as it is and change the opinion of word at 0 from neg to lneg. Consider an example for \(n=2\).

Sentence: This is sometimes[dec] a bad[neg] product.

Sentence after enriching tags: This is sometimes[dec] a bad[lneg] product.

- \([-n, +inv] \& [0, +pos] \Rightarrow [-n, +ineg, +neg, -pos, -inv] \& [-(n-1), +ineg, +neg, -lpos, -pos, -vpos, -Opn_positive] \& \ldots \ldots \& [0, +ineg, +neg, -lpos, -pos, -vpos, -Opn_positive]\)
This set of rules state that, if the word at position 0 has a positive word and the word at –n is from the list of words that invert the opinion, we remove the tags pos, inv and add the tags ineg, neg to word at –n, remove lpos, pos, vpos, Opn_positive tags if any and add ineg and neg tags to all the words from position (n-1) to position 0. Consider an example for n=5.

Sentence: I do not[inv] think this is a good[pos] product.


- \([-n, +inv]&[0, +neg]=>-[n, +ipos, +pos, -neg, -inv]&[-(n-1), +ipos, +pos, -lneg, -neg, -vneg, -Opn_negative]&………..&[0, +ipos, +pos, -lneg, -neg, -vneg, -Opn_negative]\)

This set of rules state that, if the word at position 0 has a negative word and the word at –n is from the list of words that invert the opinion, we remove the tags neg, inv and add the tags ipos, pos to word at –n, remove lneg, neg, vneg, Opn_negative tags if any and add ipos and pos tags to all the words from position (n-1) to position 0. Consider an example for n=5.

Sentence: I do not[inv] think this is a bad[neg] product.


These rules developed are used to consider phrases in the sentences (context is thus considered) and provide accurate opinions to each word. Thus, if the current word is a negation word like ‘not’, opinions of all the adjectives next to it are inverted. If the current word is from words list that increase the opinion, then the opinion of the words
next to it is changed from pos to vpos or neg to vneg. Similarly, if the current word is from words list that decrease the opinion, then the opinion of the words next to it is changed from pos to lpos or neg to lneg. This step thus results in accurate opinion tagging to each word. Consider the same input used in previous step. Following is the output after enriching tags.

- **Sentence 1**: This/[POS_DT]|Stm_this| phone/[POS_NN]|Stm_phone| is/[POS_VBZ]|Stm_be| good/[POS_JJ]|Stm_good|Opn_positive|pos| and/[POS_CC]|Stm_and| easy/[POS_JJ]|Stm_easy|pos| to/[POS_TO]|Stm_to|pos| use/[POS_VB]|Stm_use|pos|.

- **Sentence 2**: The/[POS_DT]|Stm_the| maps/[POS_NNS]|Stm_map| are/[POS_VBP]|Stm_be| very/[POS_RB]|Stm_very|inc| helpful/[POS_JJ]|Stm_helpful|Opn_weak_positive|pos|vpos| in/[POS_IN]|Stm_in| searching/[POS_VBG]|Stm_search| new/[POS_JJ]|Stm_new| places/[POS_NNS]|Stm_place|.

- **Sentence 3**: But/[POS_CC]|Stm_but| it/[POS_PRP]|Stm_it| is/[POS_VBZ]|Stm_be| difficult/[POS_JJ]|Stm_difficult|neg| to/[POS_TO]|Stm_to| carry/[POS_VB]|Stm_carry|.

The word ‘helpful’ is changed from pos to vpos, since the word ‘very’ is from the list of words that increase the opinion.

### 3.1.5 Sentence-level Opinion Mining

In this step, the opinion values are aggregated at the sentence-level and displayed beside each sentence in the input. The values of opn_positive, lpos, pos, vpos, opn_negative, lneg, neg and vneg are initially considered as 0. Then, lpos or lneg are replaced with +1 or -1 respectively. The terms pos or neg are replaced with +2 or -2
respectively. Also, vpos or vneg are replaced with +3 or -3 respectively. If the word(s) before a word also has the same opinion tag as the current word, it is counted only once and considered as a single opinion expression. We developed the following formulae to calculate the opinion value of the total sentence. +1 in the denominator is used to avoid a result of 100% positive or negative.

\[
\text{result} = \frac{\text{pos} \times 100}{\text{pos} + \text{neg} + 1}; \quad \text{if pos} > \text{neg},
\]

\[
\text{result} = \frac{-\text{neg} \times 100}{\text{pos} + \text{neg} + 1}; \quad \text{if pos} < \text{neg}
\]

\[
\text{result} = 0; \quad \text{if pos} = \text{neg}.
\]

+(0 – 100) denotes a positive sentence and –(0 – 100) denotes a negative sentence.

Consider the same input used in previous step. Following is the output after sentence-level opinion mining.

- **Sentence 1**: This phone is good and easy to use. [80%]
- **Sentence 2**: The maps are very helpful in searching new places. [75%]
- **Sentence 3**: But it is difficult to carry. [-66%]

For sentence 1, pos=4 (2 for good and 2 for easy to use), neg=0, result=(4*100)/(4+0+1)=80%. For sentence 2, pos=3 (3 for helpful), neg=0, result=(3*100)/(3+0+1)=75%. For sentence 3, pos=0, neg=2 (2 for difficult), result=(2*100)/(0+2+1)=-66%.

### 3.1.6 Document-level Opinion Mining

In this step, the opinion values are aggregated at the total document-level and displayed in the slider bar. This is done using the following formulae that we developed.

\[
\text{all\_result} = \frac{\text{all\_pos} \times 100}{\text{all\_pos} + \text{all\_neg} + 1}; \quad \text{if all\_pos} > \text{all\_neg},
\]

\[
\text{all\_result} = \frac{-\text{all\_neg} \times 100}{\text{all\_pos} + \text{all\_neg} + 1}; \quad \text{if all\_pos} < \text{all\_neg}.
\]
all_result = 0; // if all_pos=all_neg

where, all_pos is the sum of ‘pos’ values of all the positive sentences and all_neg is the
sum of ‘neg’ values of all the negative sentences in the input document. If the final value
is >= 60, it is said to be strong positive and if it is <= -60, it is said to be strong negative.
If the final value ranges among (20-60), ((-20)-20) and ((-20)-(-60)), it is said to be
positive, neutral and negative respectively. The final document-level opinion for the same
input used in the previous step is displayed with the help of a slider bar. It is shown as
70% in Figure 3.5.

![Figure 3.5. Document-level Output View](image)

For the input considered, all_pos=7 (4 for sentence 1 and 3 for sentence 2),
all_neg=2 (2 for sentence 3), all_result=(7*100)/(7+2+1)=70%.

### 3.2 Environment

The proposed system is implemented in Java. Java Swing and Jsoup parser are
mainly used. The programming environment used is Netbeans IDE, for programming
convenience.

#### 3.2.1 Java Swing

Java Swing is a Graphical User Interface (GUI) toolkit released by Oracle [29]. It
allows programmers to create GUI for their java applications. It is more sophisticated
than Abstract Window Toolkit (AWT). Swing components are completely written in
Java, unlike AWT components. Therefore, Java Swing components are platform-
independent. Its components are said to be light-weight due to high level of flexibility. Swing provides many advanced components such as lists, tables, scroll panes and tabbed panels in addition to the familiar components such as labels, checkboxes and buttons. It also provides drag and drop features for some of its components. The javax.swing package contains all the Swing classes and components. All its class names begin with ‘J’ such as JList, JButton and JFrame. Few top-level classes in Swing are said to be heavy-weight, as they extend the AWT versions. JFrame, JApplet, JWindow and JDialog are such top-level Swing classes.

3.2.2 Jsoup

Jsoup is a Java Hyper Text Markup Language (HTML) Parser. It is an open source project. Jsoup.jar version 1.7.2 is used in this project to scrape and parse the HTML content from Amazon URLs (inputs given). It is a Java library used to work with real-world HTML [30]. It provides convenient Application Programming Interface (API) to extract and manipulate data using Cascading Style Sheets (CSS) selectors, Document Object Model (DOM) traversal and jquery-like methods.

3.2.3 NetBeans IDE

NetBeans is an Integrated Development Environment (IDE) used to easily develop desktop, web and mobile applications mainly in Java [31]. It also provides tools for PHP, C/C++ and HTML5 languages. It is an open source cross-platform IDE. NetBeans IDE can run on any operating system platforms that support a compatible Java Virtual Machine (JVM) since it is written in Java. NetBeans IDE 7.3 is the latest version and is used to implement this project. NetBeans platform provides features such as User Interface management, project management, window management, storage management
and NetBeans Visual library. It also supports latest Java enhancements such as JDK7, JavaEE6 and JavaFX2. The NetBeans Editor indents lines automatically, matches brackets and words. It also highlights source code semantically and syntactically. The NetBeans Debugger allows users to place breakpoints in the source code and monitor the execution.

### 3.3 Software Modules

- Input handler - deals with two types of inputs
- Pre-processing module – deals with POS tagging, stemming and chunking
- Opinion tagging module (word-level opinion mining)
- Sentence-level opinion mining module
- User Interface module

The User-Interface module and input handler are shown in Figure 3.6. Text can be directly entered in the text area above slider bar, or a text file can be given as input, or a URL from Amazon can be entered. Text area below the slider bar shows the sentence-level opinion mining output and slider bar shows the total document-level opinion mining output.
Figure 3.6. User-Interface showing Input Handler
4. IMPLEMENTATION AND RESULTS

4.1 Reviews Entered by User as Input

Figure 4.1 shows the reviews directly entered by the user in the text area as input.

Figure 4.1. Reviews Entered by the user as Input
Figure 4.2 shows the sentence-level opinions in the text area below the slider bar and the overall output in the slider bar.

![Figure 4.2. Output for the Reviews Entered by the User](image)

This is a good product. It is very interesting. It is easy to use.
The intermediate outputs of the first four steps mentioned in the section 3.1: stemming, POS tagging, word-level opinion tagging and enriching tags, during backend processing of the input entered are shown in Figure 4.3. This is seen when the ‘See Log’ tab is clicked.

Figure 4.3. Backend Processing of the Reviews Entered by the User
4.2 Text File with Customer Reviews as Input

Figure 4.4 and Figure 4.5 show the user selecting a text file with reviews saved on desktop as input.

Figure 4.4. User Selecting Text File as Input
Figure 4.5. Text File given as Input
Sentence-level and document-level opinion output of the text file given as input is shown in Figure 4.6.

![Figure 4.6. Output for the Text File Input](image)
The intermediate outputs of the first four steps during backend processing for each sentence in the input text file are shown in Figure 4.7.

Figure 4.7. Backend Processing of the Text File Input
4.3 Amazon Webpage (URL) with Customer Reviews as Input

Figure 4.8 shows the webpage from Amazon considered as input. This page has 48 customer reviews regarding ‘Learning SQL’ textbook.
Figure 4.9 shows the URL of the webpage shown above entered as input.
Sentence-level and document-level opinion output of the given input is shown in Figure 4.10.

Figure 4.10. Output for the URL Input Entered
The intermediate outputs of the first four steps during backend processing for each sentence in the input given are shown in Figure 4.11.

Figure 4.11. Backend Processing of the URL Input
5. TESTING AND EVALUATION

A total of 100 test cases are provided to the developed system as inputs, which included customer reviews regarding iPhone5, Galaxy S3, Canon camera, Database textbooks etc in Amazon. 78 out of 100 test cases considered are actually positive according to the star rating (4 or 5) provided by the customers and the remaining 22 test cases are actually negative which has star rating of 1, 2 or 3. 72 out of 78 positive cases were shown as positive by the developed system, where as 3 cases were shown as neutral (~ False Negatives/FN) and the other 3 cases were shown as negative (FN). These wrong results are probably due to some wrongly spelt words such as ‘gud’ instead of ‘good’, ‘intresting’ instead of ‘interesting’ etc, which could not be processed by this system. 12 out of 22 negative cases were shown as negative by the developed system, where as 3 cases were shown as neutral (~False Positives/FP) and the other 7 cases were shown as positive (FP). These wrong results are probably due to the sentences such as ‘are these reviews helpful?’ etc, where the word ‘helpful’ is considered as a positive word. This problem could be solved by using better pre-processing. The evaluation information is represented in tabular form, shown in Table 5.1. All these test cases are saved as text files and the evaluation report is represented in an excel worksheet. This is shown in Figure 5.1.

Table 5.1 Evaluation Report

<table>
<thead>
<tr>
<th>Manual(Amazon)/System Output</th>
<th>Positive</th>
<th>Neutral</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>72 (TP)</td>
<td>3 (~FN)</td>
<td>3 (FN)</td>
</tr>
<tr>
<td>Negative</td>
<td>7 (FP)</td>
<td>3 (~FP)</td>
<td>12 (TN)</td>
</tr>
</tbody>
</table>
Accuracy of this system developed is calculated by using the following formula.

\[
\text{Opinion Extraction Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]

\[
= \frac{72 + 12}{72 + 12 + 10 + 6}
\]

\[
= 84\%
\]

The evaluation results in one of the recent existing systems ‘SO-CAL’ state that its average accuracy across all domains is 71.1% (on sentences, not full texts) [19]. Also, the accuracy of the existing system ‘Review Classification using SentiWordNet’ is said to be 68.63% [18]. This system used customer reviews regarding 5 products, Canon camera, Nikon camera, Nokia phone, MP3 player and DVD player in their data set. The same data set is tested with our approach and opinions of all the five test cases were classified
correctly (compared with the opinion noted by reading the reviews manually). Since, we also used Amazon reviews across various domains such as textbooks, cameras, phones etc in the data set, the accuracy of our approach can be indirectly compared with the accuracy of the two existing systems mentioned above. Therefore, the accuracy of this enhanced feature-based sentiment analysis system (84%) is greater than the accuracy of recently developed sentiment analysis systems. Thus, the opinion extraction accuracy is improved.
6. CONCLUSION AND FUTURE WORK

The feature-based sentiment analysis system developed, handles two types of input: text files and Amazon URLs, solves complex opinion structures, handles negation and opinion intensification cases. This system performs sentiment analysis at word-level, sentence-level and document-level. This is done in six main steps: stemming, POS tagging, word-level opinion tagging, enriching tags, sentence-level opinion mining and document-level opinion mining. The resources mainly used to develop this system are Stanford tool for stemming and POS tagging, SentiWordNet for enhancing the word lists and Jsoup for parsing the HTML content scraped from Amazon URLs. Also, this system achieved an accuracy of 84% which is good, compared to the accuracies of the recently developed systems such as SO-CAL and Review Classification using SentiWordNet.

Apart from the work done towards this system, future work mainly comprises of the following objectives.

- To maintain balance between speed and accuracy when better pre-processing is included.
- To improve speed when dealing with huge number of sentences.
- To test the system with huge dataset.
BIBLIOGRAPHY AND REFERENCES


43


[28] [http://sentiwordnet.isti.cnr.it/](http://sentiwordnet.isti.cnr.it/)


[31] [https://netbeans.org/features/](https://netbeans.org/features/)