An Inheritance-Based
Lexical Approach
to
Sentiment Analysis

Fathima Sharmila Satthar

A thesis submitted in partial fulfilment of the requirements of the University of Brighton for the degree of Doctor of Philosophy

October 2018
Abstract

Sentiment analysis is defined as a computational study of people’s beliefs and opinions regarding entities and events, and their attributes, as expressed in a text. The two main approaches to sentiment analysis are machine learning and lexicon-based. The machine learning approach builds a model by learning from observed data to analyse the sentiment of a text, whereas the lexicon-based approach associates sentiment scores with individual words and calculates an overall sentiment score for the document.

Each approach has strengths and weaknesses: the lexicon-based approach can capture specific lexical sentiment behaviour very precisely, but relies on expert development of lexicons which is expensive and does not scale easily; the machine learning approach can exploit broader linguistic context and so is more robust, but it is less precise and requires large-scale training data. This project introduced a novel ‘extended’ lexical approach which uses inheritance-based techniques to represent both lexical behaviour and broader linguistic context derived from corpus-based learning. This approach used lexical items not just in isolation, but in context, which allowed the study to take into account more complex linguistic constructions. The corpus-based learning technique was then used to refine this model with examples derived from corpus data. This was done by using a non-monotonic, inheritance-based architecture to represent both the lexical algorithmic component and the example-based refinements. This thesis introduced a sentiment modelling system called Galadriel, based on the inheritance mechanisms of the lexical knowledge description representation language DATR. The Galadriel system handles sentiment phrases and supports exceptions to general rules using corpus-based learning methodology. However, I did not aim to explore automatic acquisition for sentiment analysis using machine learning methods in this thesis.

More specifically, this project developed a final system (Galadriel) to address different levels of sentiment analysis related to the current research area: document-level, sentence-level and aspect-level. The main properties of the Galadriel system involve the calculation of sentiment magnitude and the polarity of a text. A cali-
bration method was introduced to assign cut-off values for the Galadriel score for each sentiment category, such as positive, negative and neutral, or for more than three-scale categories, using corpus-based learning evaluation techniques. Sensitivity and stability of the numerical position of individual lexical entries’ magnitude were then tested. This project also explored the neutral behaviour of sentiment and proposed a method to define the neutral category in sentiment analysis; the neutral class is not often addressed in the existing literature. Finally, the performance of the system was measured using precision, recall and f-score values. The evaluation results show that the Galadriel system yields comparable results across the different levels of sentiment task. The final evaluation shows that the f-score of the Galadriel system at sentence-level is 0.8284, document-level is 0.78 (three class)/0.75 (four class) and aspect-level is 0.8079(Restaurant)/0.7464(Laptop).
Contents

Acknowledgements xv
Declaration xvi
Publications and Conferences xviii

1 Introduction 1
1.1 Background and Motivation ................................. 2
1.2 Related Research Fields .................................. 4
1.3 Domains and Applications ................................ 6
1.4 Challenges .................................................. 9
1.5 Project Overview ............................................ 10
1.6 Research Strategy .......................................... 12
1.7 Contributions to Knowledge ............................... 13
   1.7.1 Theoretical Contributions .......................... 13
      1.7.1.1 Contextual Semantic Knowledge ............ 13
      1.7.1.2 Conceptual Semantic Knowledge .......... 14
      1.7.1.3 Sentiment Idioms ............................ 14
   1.7.2 Methodological Contributions ...................... 15
1.8 Thesis Structure ........................................... 15

2 Literature Review 17
2.1 Sentiment Analysis ........................................ 17
   2.1.1 Sentiment Subjectivity .............................. 18
   2.1.2 Sentiment Polarity .................................. 18
   2.1.3 Sentiment Magnitude ................................ 19
2.2 Sentiment Analysis Methodology: Previous Approaches .. 20
   2.2.0.1 Feature Selection ................................ 21
   2.2.0.2 Sentiment Lexicon ............................... 22
   2.2.1 Machine Learning Approaches .................... 23
      2.2.1.1 Supervised Machine Learning .............. 24
List of Figures

1.1 What is sentiment analysis? .................................................. 2
1.2 Some examples of related research areas of sentiment analysis .... 5
1.3 Some examples of the outcome of sentiment analysis applications . 7
1.4 Thesis structure ................................................................. 16

2.1 Sentiment analysis methodology ........................................... 20
2.2 Supervised machine learning approach to sentiment analysis .... 24
2.3 Unsupervised machine learning ............................................ 30
2.4 Lexicon-based approach methodology .................................. 34

3.1 The abstract node VERB is defined by DATR. Source: Evans and Gazdar [1996] ........................................................... 49
3.2 The morphology of the word love as defined by DATR. Source: Evans and Gazdar [1996] ......................................................... 51
3.3 The irregular behaviour of verbs as defined by DATR. Source: Evans and Gazdar [1996] ......................................................... 52
3.4 In ELF, words have access to the information of their neighbouring words. Source: Evans [2013] .................................................. 53
3.5 In ELF, The determiner A is over-ridden based on its next word. Source: Evans [2013] ................................................................. 54

4.1 Set L is divided in to subsets S and N and explained in an inheritance structure ................................................................. 60
4.2 Two further subsets of S are created ...................................... 61
4.3 The skeleton of Galadriel 0.1 framework ................................ 62
4.4 Simple sentiment model: adds up raw sentiment score of all words and produces total sentiment score ......................................... 63
4.5 Sentiment model with intensifiers: very changes sentiment score of following word ............................................................. 63
4.6 Sentiment model with negation word; not changes neg-context and changes sentiment score of following words .......................... 65
4.7 The lexical items are structured into a tree using abstract nodes
4.8 The NEUTRAL node and its subclasses in the inherited hierarchy
4.9 Static lexical information is represented and inherited using DATR
4.10 The Galadriel input and output text for a Yahoo restaurant review

5.1 Simple sentiment model: add up raw sentiment score of all words
5.2 Model sent2 for intensifiers inheriting from model sent1 with extended rule
5.3 The intensifiers are structured in a hierarchy in the Galadriel base model
5.4 The Galadriel code for the model sent2 with additional rules to handle intensification
5.5 Model sent3 for negation: the score is adjusted by neg-context
5.6 The Galadriel code for the model sent3 with the additional rule for negation
5.7 Model sent4: block-context changes sentiment scores to 0
5.8 The Galadriel code for calculating score model sent4 by considering irrealis blocking
5.9 Model sent5: changes sentiment score of the word, dependent on its word count
5.10 Model sent6: changes the total score by reducing the total score by 50% if it is negative
5.11 The Galadriel code for the sent5 and sent6 models
5.12 A summary diagram of the Opinion Observer method
5.13 OO lexicon structures in a hierarchy in Galadriel
5.14 The algorithm used to assign the value for feature found ASPECTi
5.15 The Galadriel code for assigning the score-ASPECTi values for a word
5.16 sent1 model: assigning SO values to the targeted features calculates the total feature score
5.17 The sent2 model has negation rules and re-calculates the SO value of lexical items
5.18 The sent3 model handles context-dependent words using local information
5.19 The Galadriel code for sent3 model
5.20 The sent4 model calculates total Aspect for each aspect by using the OO equation
5.21 The Galadriel code for model the OO equation in sent4
5.22 $k$-fold class mixtures, to produce PR curves for each cut-off candidate ...
5.23 PR curves for all candidate cut-off values ........................................ 118
5.24 Most appropriate PR curves .............................................................. 119
5.25 Average of Precision and Recall values ............................................. 119

6.1 The NEUTRAL node of the integrated Galadriel lexicon with updated feature values .......................................................... 128
6.2 The Galadriel code for the POSITIVE and NEGATIVE nodes ............ 129
6.3 Positive lexical items in the hierarchy .................................................. 130
6.4 Negative lexical items in the hierarchy ................................................ 131
6.5 The feature values with small font size are passed down through the hierarchy and the feature values with large font size are overridden in the Galadriel base model .................................................. 132
6.6 The Galadriel code for some positive words modelled in the Galadriel (lexicon) base model .............................................................. 133
6.7 Context words show sentiment depending on their context .................. 134
6.8 Context words can have different magnitude values ........................... 134
6.9 The Galadriel code: the lexical items neighbouring and and ‘,’ are checked for if they contain same part-of-speech tag ......................... 136
6.10 Interrogative lexical items ................................................................... 137
6.11 Some words and phrases that have irregular sentiment behaviour, so they can be defined in a different class ........................................... 138
6.12 Listing the phrase in training data with its frequency ........................... 139
6.13 Phrases are modelled in the base model ............................................. 140
6.14 Sentiment phrases are modelled in the base model by considering one or two lexical items of the phrase ................................................. 142
6.15 The integrated Galadriel sent1 model with its features ......................... 143
6.16 Galadriel code for assigning context-pol values for words in a sentence in the sent1 model .............................................................. 145
6.17 The features handle context dependent words in Galadriel model-2 ........ 147
6.18 The Galadriel code: the context-pol value is calculated in the sent2 model ................................ ......................................................... 148
6.19 Intensifiers recalculate the magnitude values in Galadriel model 3 ........ 148
6.20 The negation rules are applied to the sent4 model followed by a negator 149
6.21 Irrealis rules are applied to words that come after an irrealis word in the model ................................................................................. 150
6.22 The rules handling interrogative sentences in Galadriel model sent6....... 151
6.23 Aspect class terms are structured in the hierarchy ............................... 161
6.24 Total sentiment score for each of the targeted aspects is calculated ....... 163
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Some lexical entries with their semantic orientation according to different lexicons</td>
<td>19</td>
</tr>
<tr>
<td>2.2</td>
<td>Summary of some previous supervised learning approaches</td>
<td>29</td>
</tr>
<tr>
<td>2.3</td>
<td>Summary of some previous unsupervised learning approaches</td>
<td>33</td>
</tr>
<tr>
<td>2.4</td>
<td>Summary of some previous lexicon-based approaches for sentiment analysis</td>
<td>37</td>
</tr>
<tr>
<td>2.5</td>
<td>The outcomes of the binary classification formulated in a confusion matrix</td>
<td>41</td>
</tr>
<tr>
<td>5.1</td>
<td>Summary table of SO-CAL features and their calculation</td>
<td>79</td>
</tr>
<tr>
<td>5.2</td>
<td>Performance of SO-CAL and Galadriel models for only adjective and all words</td>
<td>90</td>
</tr>
<tr>
<td>5.3</td>
<td>Comparison of the performance of SO-CAL features and Galadriel models (all words)</td>
<td>90</td>
</tr>
<tr>
<td>5.4</td>
<td>Comparison of the performance of SO-CAL and Galadriel on positive and negative reviews</td>
<td>91</td>
</tr>
<tr>
<td>5.5</td>
<td>Summary table of OO features and their calculation</td>
<td>95</td>
</tr>
<tr>
<td>5.6</td>
<td>Comparing the performance of OO and Galadriel</td>
<td>109</td>
</tr>
<tr>
<td>5.7</td>
<td>Overall f-score of OO and Galadriel models</td>
<td>109</td>
</tr>
<tr>
<td>5.8</td>
<td>Average precision, recall and f-score measures for candidate cut-off values</td>
<td>120</td>
</tr>
<tr>
<td>5.9</td>
<td>Cross-validation process for calibration system</td>
<td>120</td>
</tr>
<tr>
<td>5.10</td>
<td>Confusion matrix for the classification</td>
<td>121</td>
</tr>
<tr>
<td>5.11</td>
<td>Comparing performance measures calculated by the calibrated and uncalibrated versions of Galadriel</td>
<td>121</td>
</tr>
<tr>
<td>5.12</td>
<td>Performance comparison of Galadriel with calibration method on SO-CAL features</td>
<td>123</td>
</tr>
<tr>
<td>5.13</td>
<td>Performance (f-score) comparison of Galadriel with calibration method on OO features</td>
<td>124</td>
</tr>
</tbody>
</table>
6.1 Comparison performance integrated Galadriel with SO-CAL feature on positive and negative reviews .................................. 153

6.2 Evaluation results of the Galadriel system on the STS-Gold dataset using both uncalibrated and calibrated evaluation methods along with the Maximum Entropy classifier used by Saif et al. [2013] .................................. 155

6.3 Evaluation result comparison between Galadriel and other systems on STS-Gold dataset .................................................. 155

6.4 Evaluation result comparison between Galadriel with both calibrated and uncalibrated evaluation methods and the baseline methods .......................................................... 157

6.5 Evaluation of overall performance measures of three-class and four-class classification of Galadriel without the calibrated evaluation method .......................................................... 160

6.6 Evaluation of overall performance measures of three-class and four-class classification of Galadriel with calibrated evaluation method .................................................. 160

6.7 Average accuracies comparison between Galadriel and Pang and Lee (2005)’s algorithms .......................................................... 160

6.8 Comparison performance of integrated Galadriel and basic Galadriel with OO features .......................................................... 165

6.9 Evaluation results of Galadriel’s aspect-base model (sent7) using calibrated and uncalibrated methods .................................................. 167

6.10 Comparison f-score and accuracies between Galadriel and some participations in SemEval2016 on Restaurant and Laptop Domains .......................................................... 167

6.11 Evaluation result comparison between the Galadriel neutral model and the Galadriel standard model on various domains .......................................................... 172

6.12 Experiments carried out while changing the magnitude values of the Galadriel system .......................................................... 176

6.13 Performance measures of three classes according to the different magnitude values of nodes .......................................................... 178

6.14 Performance measures of the three classes according to the different magnitude values of nodes .......................................................... 178

6.15 Experiments were carried out while changing the magnitude values of the Galadriel system in five arithmetic sequences .......................................................... 180

6.16 Experiments were carried out while changing the magnitude values of the Galadriel system in five polynomial sequences .......................................................... 181

6.17 Performance measures of three classes according to the values of the magnitude nodes in arithmetic sequences .......................................................... 182

6.18 Performance measures of three classes according to the values of the magnitude nodes in polynomial sequences .......................................................... 182

6.19 Experiments were carried out while changing the magnitude values of the Galadriel system in five arithmetic sequences .......................................................... 180

6.10 Experiments were carried out while changing the magnitude values of the Galadriel system in five polynomial sequences .......................................................... 181

6.11 Performance measures of three classes according to the values of the magnitude nodes in arithmetic sequences .......................................................... 182

6.12 Performance measures of three classes according to the values of the magnitude nodes in polynomial sequences .......................................................... 182
6.19 Performance measures of four classes according to the values of the magnitude nodes in arithmetic sequences
6.20 Performance measures of four classes according to the values of the magnitude nodes in polynomial sequences
6.21 Average performance measures of three classes according to the values of the magnitude nodes in both arithmetic and polynomial sequences
6.22 Average performance measures of four-class classification according to the values of the magnitude nodes in both arithmetic and polynomial sequences
6.23 Mean average error for both three- and four-class classification
Acknowledgements

Getting a PhD was one of my dreams, and completing this PhD is an unforgettable moment which has changed my life entirely. My PhD experience has been the most exciting journey of my life. This journey would not have been possible without several people who have travelled with me along the way.

Firstly, I would like to express my sincere gratitude to my supervisor Dr Roger Evans for his continuous support of my PhD and related research, and for his patience, motivation and immense knowledge. His guidance helped me throughout all the time of researching and writing this thesis. I am also thankful to him for providing the DATR code and other technical support.

I would also like to thank my second supervisor, Dr Gulden Uchyigit, who provided help and support in this research. Her valuable feedback and advice greatly contributed to my thesis.

I would also like to use this opportunity to thank the Doctoral College at the University of Brighton for facilitating me in carrying out this research, and for funding me to attend conferences. I want to thank the School of Computing, Engineering and Mathematics at Brighton, who gave me the opportunity to work as an assistant lecturer during my PhD. I would also like to say a big thank-you to all my university friends for supporting me, and special thanks to Dr Simon Davies, who helped me in proofreading my thesis.

I would like to thank Dr Maite Taboada, who kindly gave me access to the SO-CAL system, its dictionaries and the datasets.

Finally, a very special thanks my husband, Shiraz Rafeek, for supporting and encouraging me throughout my PhD and always being there for me.
Declaration

I declare that the research contained in this thesis, unless otherwise formally indicated within the text, is the original work of the author. The thesis has not been previously submitted to this or any other university for a degree, and does not incorporate any material already submitted for a degree.

29th October, 2018
Publications and Conferences

Publications


Oral and Poster Presentation

• ICCSW15- 2015 Imperial College Computing Student workshop, London, United Kingdom. September, 2015 (Best Presentation Award)


• Poster and oral presentation- ‘Pathways to Innovation’, The life and Physical sciences postgraduate research student conference, University of Brighton, July, 2014
Poster presentation-'Making Science Accessible', Research Student Conference, University of Brighton, July 2013
Chapter 1

Introduction

"People influence people. Nothing influences people more than a recommendation from a trusted friend. A trusted referral influences people more than the best broadcast message. A trusted referral is the Holy Grail of advertising."

Mark Zuckerberg, Facebook CEO

Word of mouth makes a significant impact on the decision-making process in the real world, where people consider others’ feedback, opinions and emotions. This process has been transformed thanks to the Internet, and now people are interested in listening to the world via social media platforms, and have started to interact with the world via web videos, audio, blogs and more. As social media has become more popular and easily accessible, people have begun to approach blogs, web posts and reviews for help in making decisions. Most of them are only interested in others’ summarised opinions – yes or no? good or bad? – towards an item or a particular topic, which can be described by the term ‘sentiment’. Emerging technologies like artificial intelligence have been deployed to assist people with their decision making by extracting sentiments from digital documents. This study aimed to build a system for sentiment analysis using a novel approach by explicitly considering advanced lexical knowledge representation, which involves non-monotonic (default) inheritance networks. The approach uses lexical items not just in isolation, but in context, which allows us to take into account more complex linguistic constructions. I built models in an inheritance structure, in which each model has different rules
and techniques to handle various types of linguistics features.

1.1 Background and Motivation

Sentiment analysis is a computational study of people’s opinions of, appraisals of and emotions regarding a product, an entity or an event, or their attributes. During the decision-making process, ‘what other people think’ has always been important information. Whenever someone makes a decision, they want to get to know others’ judgements and beliefs: this is true not only for individuals but also for business organisations, who want to take their customers’ pulse. Previously, when an organisation or individual made a decision, they typically asked for the opinions of other people or their target audience by conducting a survey, and when a person was seeking others’ ideas, they asked their friends and relatives. However, in recent years, the world has been transformed with the explosive growth of innovative technologies. Now people can post their opinions and have discussions on forums and social networks, or post their reviews of products on the web or a particular organisation’s merchant site. Now, if a corporation or company wants to find out their consumers’ opinions, they do not need to conduct surveys. In order to gather customer opinions about their service/product and those of their competitors, they can refer to online reviews, as there is plenty of information available on websites. Similarly, if someone

Figure 1.1: What is sentiment analysis?
seeks information about a product/service, they do not need to call their friends or relatives anymore. However, there are a significant number and variety of sites, and each site has a large volume of opinionated text. Therefore, finding an appropriate site and locating the relevant information on the web is still a difficult task. It is challenging for a human to find particular sites and extract specific sentences with opinions, then read, summarise and organise them into usable forms, as the opinions and views are hidden in the huge volume of text on forum posts (for instance).

Computers have thus begun to be used for searching out and understanding others’ views, a process which is known as sentiment analysis. According to Mejova (2009), sentiment analysis can be defined as to extract, identify and characterise the sentiment content of a text unit using Natural Language technology, statistics and machine learning. Most sentiment analysis techniques determine the polarity of sentiment in a text – whether positive, negative or neutral. Emotion analysis is a branch of sentiment analysis, and involves detecting more specific emotions that are expressed in the text (Aman and Szpakowicz, 2007; Bhownick, 2009; Strapparava and Mihalcea, 2008). Emotion analysis is, however, out of the scope of this thesis.

Sentiment analysis involves several separate tasks. The first step is sentiment sentences detection, which is described as filtering out objective sentences, leaving behind subjective sentences that usually include all the sentiment-bearing content. Some of the earlier researchers demonstrated that this could be determined easily by looking at adjectives (Hatzivassiloglou and McKeown, 1997a) and adverbs (Benamara et al., 2007). The second step is the polarity of the classification. This identifies whether the given opinionated text is positive, negative or neutral. Another step detects the strength of the sentiment (Rosenthal et al., 2017; Lee and Grafe, 2010; Pang and Lee, 2005).

Sentiment analysis can be applied on different levels, such as the term, phrase, sentence or document levels. To illustrate, a given opinionated text could be a word, phrase, sentence or a complete document. The word-level sentiment analysis task detects the sentiment of a particular term, which could be a single term or within a particular context/sentence. Phrase-level sentiment tasks consider a given phrase and detects whether the phrase expresses sentiment or not. The sentence-level task focuses on identifying the sentiment of a given sentence. Some of these tasks also involve detecting whether a sentence is subjective or objective (Wiebe et al., 1999; Yang and Cardie, 2014). The document-level task classifies the sentiment of the overall document. Product reviews or movie reviews are examples of texts which can be used in document-level analysis (Turney, 2002; Pang et al., 2002; Dave et al., 2003; Moraes et al., 2013). This task determines whether a document is positive, negative or neutral on a single topic or a target.
Discovering the target is another important task in sentiment analysis. The target is an entity about which the opinion/sentiment is expressed. This means that it is always important to identify whether a given statement talks about the specified product. For example, most of the general writing on blogs and webpages does not provide a pre-defined topic as the target of the blog. In some approaches, target/topic/aspect detection is part of the sentiment analysis task [Balahur et al., 2013; Fahrni and Klenner, 2008; Araque et al., 2016], but I consider it as a separate process, so for this project, ‘sentiment analysis’ is relative to a particular target provided as part of the task input. Thus, identifying the topic of a document is out of the scope of this study. Aspect-level (or aspect-based) sentiment analysis tasks have become more popular over the last few years. These tasks extract the sentiment of a targeted aspect of a document or sentence. Identifying the target aspect is another complex task and also out of the scope of this study. Previously, this task was known as feature-based sentiment analysis (Hu and Liu, 2004; Ding et al., 2008).

In addition to the above tasks, entity-level sentiment analysis has become a popular task in organisations and companies. Entity-level sentiment analysis predicts the sentiment expressed about an individual entity in an aggregated number of documents or a corpus. For example, this task determines the sentiment expressed in regards to an entity such as a political party, a brand, etc. This task uses a set of documents/tweets or social media big data to detect the sentiments of each document in regards to a particular brand/party, aggregates them, and determines the overall sentiment towards the brand (Godbole et al., 2007).

1.2 Related Research Fields

Artificial intelligence (AI) is a branch of Computer Science that aims to enable computers to complete human intelligence tasks. The field of AI has been growing vigorously in recent years, to the point that it impacts on human lifestyles. Machine learning is one of the important techniques of AI. Machine learning techniques build systems for AI applications by learning from previous experiences, rules, algorithms and patterns. Natural language processing (NLP) is an area of AI which builds system for understanding human language, and most NLP applications utilize machine learning techniques. Figure 1.2 situates sentiment analysis within these areas and shows some of its related subfields. This section provides a brief overview of these related research areas.

Data mining is one of the popular applications of AI. Data mining is a computing
process of detecting patterns in large datasets, and helps companies and organisations turn raw data from different sources into useful information. Many researchers have been working on knowledge discovery using data mining techniques in recent years (Fayyad et al., 1996b, Miller and Han, 2009, Liu and Motoda, 2012, Prather et al., 1997). Machine learning methods are prominently used in the data mining process (Hall et al., 2009, Witten et al., 2016). Text data mining (or simply text mining) is a computational method of detecting information from text such as documents. However, unlike data mining, text mining is involved in more complex, unstructured text data. In addition to machine learning techniques, keyword techniques (Noh et al., 2015) and linguistic techniques (Rajman and Besançon, 1998, Kao and Poteet, 2007) are employed in text mining (Hothe et al., 2005, Aggarwal and Zhai, 2012). Researchers have used text mining in various research areas. Text mining makes use of the technology for information extraction (IE) (Ponte and Croft, 1998, Baeza-Yates and Ribeiro-Neto, 1999) which is a computing process that extracts relevant information from a given text. IE has been used in different domains for many years. For example, biomedical researchers have used text mining techniques to extract information from medical records (Rodriguez-Esteban, 2009, Cohen and Hunter, 2008). Document summarisation (Cohen and Hunter, 2008, Hu and Liu, 2004) is a popular subtask of IE, and has been widely used by business organisations to extract the main points from their customers’ reviews. Similarly, sentiment analysis or opinion mining presents the summary of a text in one word.

Figure 1.2: Some examples of related research areas of sentiment analysis
Entity extraction (or named-entity extraction) is another sub-task of IE, and identifies names of people, locations or other entities. This task is quite popular when collecting big data on a specific subject and its related information by searching the web (Etzioni et al., 2005). Researchers have employed entity extraction in different research areas. For instance, Krallinger et al. (2013) used entity recognition for drug and chemical compound name extraction; names of genes and proteins have been identified in biomedical research (Humphreys et al., 2000; Borthwick et al., 1998); and names of authors and titles have been detected from online publications (Lawrence et al., 1999; McCallum et al., 2000).

Information retrieval (IR) is another popular task in text mining. IR is a process of finding a document (information) that is relevant to an information need from a large collection of data. Web search engines are the best example of applications of IR (Croft et al., 2010). Digital libraries also make use of IR, allowing users to access the information they hold (Schatz, 1997; Witten et al., 1996; Selvam, 2014). IR techniques have also been employed in building information filtering systems such as recommender systems (Costa and Roda, 2011; Cacheda and Parapar, 2015).

Machine translation (MT) is another application of NLP that uses machine learning techniques. MT translates text or speech from one language to another, which helps to connect people from across the globe. Researchers have been working on building accurate MT systems since 1949 (Hutchins, 2000). Various approaches have been applied to MT, such as the rule-based approach (Forcada et al., 2011) and the statistical approach (Brown et al., 1990; Koehn, 2009; Lopez, 2008).

1.3 Domains and Applications

The focus of the present thesis is sentiment analysis. Sentiment analysis helps organisations and individuals monitor real-world events. This section enumerates a few real-world applications of sentiment analysis and the possible domains where they can be applied.

Product reviewing is the traditional domain in which sentiment analysis is employed. Organisations widely use sentiment analysis of their clients’ reviews and product feedback to improve their service as well as to attract new customers. Another popular domain is movie reviews. Thousands of sentiment analysis studies have been conducted on product reviews and movie reviews in recent years (for example, Shirani-Mehr (2014), Pouransari and Ghili (2014), Leung (2009), Chankarkar et al. (2012), Rain (2013)). There are numerous sentiment analysis tools available in the
market to purchase for commercial purposes, such as Lexalytics.\footnote{https://www.lexalytics.com/}

Sentiment analysis applications have been widely used on social media text, such as text from Twitter, Facebook posts, Instagram or YouTube comments, for various purposes. Social media monitoring companies such as Brandwatch\footnote{https://www.brandwatch.com/} and Crimson Hexagon Analytics\footnote{https://www.crimsonhexagon.com/} offer software services that help businesses track their brand’s online presence. Some SA applications have been used, for example, to predict election and referendum results using Twitter posts – a notable example of this was the ‘Brexit’ referendum\footnote{http://blogs.sps.ed.ac.uk/neuropolitics/2016/01/06/uk-eu-twitter-sentiment-analysis-an-analysis-of-the-sentiment-in-the-twittersphere-towards-the-uk-leaving-or-remaining-in-the-eu/}.

(a) An example product review showing overall sentiment breakdown of several aspects of the product

(b) A sentiment analysis dashboard showing overall aggregated customer sentiment towards a product over a specific period

(c) A graph showing the mean sentiment scores change over a period using Brexit tweets containing positive and negative opinions

(d) Jockers \cite{jockers2014} presented this graph showing sentiment over the progression of a narrative as a percentage of a novel

Figure 1.3: Some examples of the outcome of sentiment analysis applications

Recently, there has been a significant growth in sentiment analysis of news articles \cite{litwiniuk2016, raina2013}. However, sentiment classification of news is slightly...
different from other types of text. Three different views, such as author, readers and
text, have been taken into account in this area, and have been addressed differently
in sentiment analysis, for example by Balahur et al. (2013). Some researchers have
applied sentiment analysis applications to financial news articles to predict stock
market developments (Kalyanaraman et al. 2014; Zhai et al. 2011). Godbole et al.
(2007) developed a large-scale sentiment analysis system which extracts sentiments
on multiple entities in news and blogs. Nick Altmann created an online tool for
sentiment analysis of Guardian articles based on readers’ comments.

Some academic institutes use sentiment analysis applications to conduct course and
teaching evaluations (Pong-Inwong and Kaewmak 2016). The institutes ask stu-
dents to provide textual feedback and comments about the instructors and the course
delivery (Rajput et al. 2016; Mac Kim and Calvo 2010). Some researchers (e.g.
(Wen et al. 2014; Adamopoulos 2013)) use students’ forum posts to explore student
sentiments regarding online courses.

As well as uses in the commercial sector, analysing sentiment in legal texts and
blogs is extremely valuable to people who work on legal issues. Conrad and Schilder
(2007), for example, discuss the scope of sentiment analysis in legal blogs.

Sentiment analysis applications such as MUSE have been used to identify some-
one’s individual messages over several years by tracking sentiments in their email
archives (Hangal and Lam 2011). Many digital communication companies have al-
ready started to introduce sentiment analysis applications for SMS and smartphone
messages to the market, such as Twilio. This application automatically labels SMS
messages as positive, negative or neutral (Andriotis et al. 2014).

Applying sentiment analysis to a narrated story is another complicated task. Jockers
(2014) worked on implementing sentiment analysis tools and techniques on fiction,
and presented sentiment graphs for plot movement. Later, he developed an applica-
tion for exploring the relationship between sentiment and plot arc (Jockers 2015).
Landit (2010) also used sentiment analysis tools for understanding fiction.

The goal of a sentiment analysis application is to expose the final sentiments of a
given text in regards to a product or other targeted entity. Visualisation is an effec-
tive method for the presentation of this data. For example, commercial sentiment
analysis tools and applications produce their final sentiment results on a product or
entity through bar charts, pie charts or graphs. Then readers or clients can track
the historical and daily sentiment analysis reports of the required entries. Figure

---

1.3 shows the example output of some sentiment applications.

Additionally, sentiment analysis applications are used for other research purposes, such as modelling spoken dialogue systems (Vanrompay et al., 2014).

### 1.4 Challenges

Sentiment analysis is a popular research area, and there have been over 7000 research projects and articles written on the topic. Nevertheless, there are still many major challenges, some of which have been identified at previous years’ Sentiment Analysis Symposium, an annual conference that addresses the business value of sentiment, opinion, emotion and intent, including:

1. There is a lack of suitable modelling of compositional sentiment, which means that the overall sentence sentiment of the sentiment bearing word, the sentiment shifters and the sentence structure need to be calculated more accurately at the sentence level.

2. Sentiment lexicons are one of the important features used in sentiment analysis. Creating a sentiment lexicon is another challenging task. Building a lexicon with semantic intensity scores is extremely beneficial. However, having such scores annotated by human annotators is not feasible as it is difficult to maintain consistency across different annotators. Various lexicon sources are publicly available for sentiment analysis. However, which sources give the most reliable semantic scores has not yet been established.

3. In the same document, a product may be referred to by many names. This is one of the main issues of automatic name entity resolution, and has not yet been solved effectively. The handling of anaphora resolution in an accurate way is another major issue in text mining. It is an important, challenging issue in sentiment analysis too.

4. It is essential to identify the text relevant to each entity, when there are several entities discussed in a document. The current accuracy of the identification of relevant text does not give satisfying results.

5. Another big challenge in sentiment analysis system is handling noisy text (text with spelling/grammatical mistakes, missing/problematic punctuation, slang, etc.).

6. Handling sarcasm and irony is another challenge in sentiment analysis. Some reviews tend to show their dissatisfaction towards a product/service in a sar-
castic way using positive language. Identifying sarcasm has not yet been properly integrated within sentiment analysis systems, although some previous researchers (e.g. (Riloff et al., 2013)) have worked on recognition of sarcasm in the field.

7. A new approach is needed to handle factual statements. Many statements about factual entities carry sentiment. But only subjective statements are considered in most of the current sentiment analysis methods, and researchers fail to consider such factual (objective) statements.

8. Some authors like to use ambiguous comments in their posts. Ambiguous words and statement may be humorous but can lead to vagueness and confusion. Expressing the meaning of such statements without context is difficult.

9. In some cases, applications translate foreign customers’ reviews into English. Many translation programs have difficulty correctly interpreting sentiments in language, as Western and Asian or African sentiments differ from each other significantly.

1.5 Project Overview

This project aimed to employ lexicon representation, which can encode very complex information such as phonology, morphology, syntax and semantics, for sentiment analysis. My basic sentiment framework was modelled using an inheritance-based lexical knowledge representation language called DATR (Evans and Gazdar, 1996), which supports the notion of inheriting lexical information from abstract classes, but also the possibility of overriding inheritance. I also used the DATR extension library ELF (the Extended Lexicon Framework (see Evans (2013))), which allows DATR to support fully lexicalised models of (non-lexical) language processing (such as part of speech tagging, parsing, or, in this case, sentiment analysis). Chapter 3 provides a detailed review of the DATR lexical knowledge description language and its extended lexicon framework, ELF (Evans 2013). To design the research project, I started with a research question and then set key objectives/goals.

Research Question: Can inheritance-based modelling techniques be used to improve the modelling of sentiment in a text?

- Objective 1: To model sentiment knowledge using DATR’s inheritance mechanism by modelling existing lexicon-based approaches to sentiment analysis, evaluate the effectiveness of the model and identify scope for improvement.
• **Objective 2**: To combine and extend models of existing systems to provide an innovative rule-based system using the inheritance-based model of sentiment knowledge.

• **Objective 3**: To refine the inheritance-based model by extending and/or overriding its rule-based system based on corpus-analysis techniques.

• **Objective 4**: To evaluate the proposed model quantitatively in order to assess the effectiveness of inheritance-based modelling techniques for sentiment analysis.

In order to meet the above objectives, I tried to break them down into key questions and aimed to answer them. This section explains how I used those questions to accomplish the research objectives.

Much sentiment analysis work makes use of lexical information about the sentiment of individual words. At the simplest level, this is just a list of words which have a positive sentiment and another list of words which have a negative sentiment. Some approaches have tried to incorporate more knowledge into their lexicons, for example information about context. There has been a whole tradition of work on lexicon representation which can encode very complex lexical information, such as phonology, morphology, syntax and semantics. But this approach has not been exploited for sentiment analysis. The particular task that I undertook here was based on the language of DATR (Evans and Gazdar 1996), which supports the notion of inheriting lexical information from abstract classes, but also the possibility of overriding inheritance. So, first I tried to answer the following questions:

• **Q1**: Can recent work on lexicon-based sentiment analysis be modelled using DATR?

• **Q2**: Is there an advantage to using DATR’s inheritance mechanisms to model sentiment knowledge?

• **Q3**: Can sentiment be inherited? (Either as an extension to existing approaches or as a new approach.)

One of the directions of current developments regarding DATR is ELF (Evans 2013), the extended lexicon framework. In ‘ordinary’ DATR, the language is used to represent information about individual words, and it is assumed that there is some system outside that wants to make use of this information. ELF uses DATR to represent words not as isolated individuals, but as instances occurring in sentences. The information is still represented on a word-by-word basis, but the stored information about a word can include information about its neighbours in a sentence. So, the next set of questions were:
Q4: Can I extend an inheritance-based model of sentiment knowledge of words to a model of sentiment analysis? Can sentiment behaviour be inherited?

Q5: Can I encode the entire sentiment analysis task as a ‘lexical description’ task using ELF?

Q6: Can I then use DATR’s ability to encode exceptions to obtain a very fine-grained model of how sentiment works, which would be more accurate?

Q7: Can I use corpus-based learning methodology to populate such a model with examples derived from corpus data?

I addressed objectives 1 to 4 by answering these questions with appropriate evidence. Finally, I developed the final system, Galadriel version 1.0.

1.6 Research Strategy

My research strategy sets out a plan for exploiting inheritance-based lexical representation for sentiment analysis. Inheritance is the result of reasoning over the paths in a hierarchy. A key benefit of inheritance is to minimise the amount of duplicate information in multiple subclasses by re-factoring common information to a mutual super class, which provides a better organisation of rules. Moreover, there is more flexibility in changing the rules in the inheritance when using the superclass/subclass interchangeably. Non-monotonic inheritance is a default inheritance, in which the classes can be replaced or overridden, and it allows for exceptions. Non-monotonic inheritance reasoning has several benefits, such as re-usability, extensibility, overriding, etc. Therefore, I aimed to exploit non-monotonic inheritance reasoning techniques for modelling sentiment analysis, and thus this thesis addresses the following research question:

Can inheritance-based modelling techniques be used to improve the modelling of sentiment in a text?

DATR/ELF provided a framework that supports fully lexicalised sentiment analysis, that is, it calculates sentiment purely on the basis of interactions between lexical items in a sentence. My starting point was a very primitive sentiment analysis system implemented in that framework and named as Galadriel 0.1. On top of this I modelled two existing and contrasting sentiment analysis systems, in order to develop modelling techniques and validate the approach. Then I gradually developed a system, Galadriel 1.0, using elements from these models, plus additional insights and extended word and phrasal lexicons capturing detailed exceptional lexical behaviour, derived from corpus analysis. The complete, developed Galadriel system
addresses different levels of sentiment analysis related to the current research area: document-level, sentence-level and aspect-level. The Galadriel system calculates sentiment scores by combining raw lexical scores using a range of arithmetic rules (summing, scaling, averaging, etc.). The final output of Galadriel for a text is a signed real number which reflects sentiments expressed by the lexical items in the text and the syntactic and semantic relationships between them. Finally, I evaluated Galadriel against various gold standard sentiment analysis methods. I proposed a pre-evaluation process that calibrates Galadriel’s final results with the others in order to optimise the mapping from Galadriel’s score to the sentiment classes specified in the gold standard data.

1.7 Contributions to Knowledge

This section discusses my contributions to knowledge. Exploiting inheritance-based lexical knowledge for sentiment analysis, which has not been explored before, is my major contribution. Moreover, I divide my novel contributions to the modelling approach to sentiment analysis into two sub-topics, theoretical and methodological.

1.7.1 Theoretical Contributions

The theoretical contribution of this thesis is the modelling of a sentiment lexicon in an inheritance structure based on the sentiment behaviour of lexical items. The top-level abstract node of the hierarchy has a default definition which is inherited to the children nodes, and relevant rules override exceptions. The lexical items are placed in the lower level of the hierarchy, and their definitions are inherited from their abstract node, which makes the process much easier using the extension and the adaptation of lexicon.

Moreover, the non-monotonic inheritance mechanism allows us to model contextual and conceptual semantic knowledge to the lexical items. Additionally, it enables us to define the lexical items that have irregular sentiment behaviours, such as sentiment idioms/phrases.

1.7.1.1 Contextual Semantic Knowledge

Some lexical items express a different meaning than their lexical semantics. Consider the lexical unit good in the following sentences:

1. ‘This is a good place to work.’
2. ‘I have left this company for good.’

The lexical unit *good* in the above sentences does not express the same sentiment in each instance. My theoretical framework allows the lexical item to access information about its neighbouring lexical items and define the sentiment behaviour according to this contextual information. For example, in the first sentence, *good* behaves as a positive sentiment word. However, in the second sentence, *good* changes its behaviour because it is preceded by *for*.

Another contribution is the introduction of a method for handling context-dependent sentiment behaviour of lexical items. Some descriptive adjectives, such as *long* and *short*, are used in statements to express a sentiment indirectly. I proposed a method that extracts the author’s sentiment from such statements by considering the author’s state of mind.

### 1.7.1.2 Conceptual Semantic Knowledge

Particularly in aspect-level sentiment tasks, some words do not directly refer to the targeted aspect. For example, consider the following reviews from the restaurant domain:

1. ‘The lamb dish was small and frankly not very exciting.’
2. ‘The fillet steak was OK.’

The terms *lamb* and *steak* are used to refer to *food* (assumed to be the targeted aspect) in the aspect-level task. I populated those terms using a training corpus and modelled them under an abstract node in the sentiment lexicon, which provides their sentiment behaviour by inheritance.

### 1.7.1.3 Sentiment Idioms

Sentiment phrases and idioms are groups of lexical items which expresses sentiment regardless of individual words’ meaning. For example:

1. ‘The bee’s knees’
2. ‘Kiss of death’

The above idioms express positive and negative sentiments, regardless of the meaning of the individual words in the idioms. These idioms are modelled in the sentiment lexicon using the inheritance-based mechanism.
1.7.2 Methodological Contributions

I developed a novel framework (*Galadriel*) for fine-grained sentiment analysis, a system with a base model where lexical items are modelled in an inheritance-based lexicon with a raw semantic score based on their sentiment behaviour, and a model that calculates the sentiment score of lexical items by different arithmetic rules.

Syntax changes the sentiment behaviour of lexical items, as well as their sentiment score. I added another set of models to the framework that re-calculate the scores of lexical items in grammatical structures. Each model has different rules and algorithms to handle various linguistic features, and one model inherits from the other. This modelling approach is one of my methodological contributions to knowledge. The inheritance modelling structure allows the replication of different techniques in the same framework.

A further contribution of this thesis is the proposal of a definition for a neutral sentiment class, distinguished from the mixed sentiment class. I introduced an added model to the *Galadriel* framework that can extract positivity and negativity from a text separately. The model also detects simple neutral-class text.

Finally, this thesis also introduced a calibration method that maps sentiment numerical scores to sentiment classes, which gives scoring sentiment analysis systems the ability to produce standard sentiment classes.

1.8 Thesis Structure

This thesis is structured in seven chapters. Those chapters are organised as shown in figure [1.4]. In the first chapter (this chapter), I discuss the background to and motivation for this thesis, along with the research strategy, research question and objectives. Chapter 2 includes a literature review, with a discussion of previous research methodologies, evaluation methods, and the research gap. Chapter 3 provides a review of the technical tools that are used for this work, namely DATR, a language for lexical representation, and ELF, the extended lexicon framework. Chapter 4 introduces a novel modelling framework, *Galadriel*, based on inheritance structure, and outlines its key features. Chapter 5 covers a representative lexicon-based approach. I studied two different existing lexicon-based approaches and modelled their features in the *Galadriel* framework. I run the *Galadriel* system using the same datasets that have been used in the existing systems and validate *Galadriel* by comparing its results with the results of the original systems. I also introduce a novel technique to calculate the evaluation matrix in this chapter. Chapter 6 develops an
integrated sentiment analysis system by merging the techniques of the existing lexicon approaches and adding new techniques. Chapter 6 also includes an evaluation of the integrated system. Finally, Chapter 7 presents a summary, conclusion and suggestions for future work.
Chapter 2

Literature Review

This chapter provides a review of previous studies in the sentiment analysis research area. I start with a discussion of how sentiment analysis has been defined by various researchers. I then discuss various approaches that have been used in the field. I also present a discussion of the various inheritance models that have been used. I then explore evaluation techniques that have been used to evaluate different methods. I also introduce some gaps that I have identified in previous research. Finally, I discuss the research methodology.

2.1 Sentiment Analysis

Sentiment analysis is a process of determining the opinion or perception of a piece of text. A human can easily detect whether a text displays a positive or negative opinion by looking at the language it uses. Although researchers have begun to look at sentiment analysis in a big way in recent years, there has always been an undercurrent of interest in the area. Henken (1976) used a computer to analyse context in text documents in suicide and murder investigations. This was one of the earliest studies investigating a written document via a computer to find out the author’s sentiment. Then researchers began to focus on modelling authors’/speakers’ beliefs for computer understanding of natural language (Wilks and Bien, 1984; Carbonell 1981). During the same period, the field of information retrieval started to become more popular (Furnas et al., 1988; Dumais et al., 1988). Later projects began to focus on more areas such as point of view, affect and emotion, meta linguistic facts, narrative, evidentiality in the text, interpretation of metaphor, subjectivity, and related areas (Wiebe and Rapaport, 1988; Wiebe, 1990; Hearst, 1992; Wiebe, 1994; Subasic and Huettner, 2000; Kantrowitz, 2003).
At the end of the twentieth century, a widespread awareness of research problems, questions and activities regarding sentiment and opinion mining was raised. Business and social science researchers had been using the sentiments and opinions of groups of people, markets or organisations in their research in order to predict the final results or returns of investment (Lee et al., 1991; Carroll et al., 1994; Eichengreen and Mody, 1998). Researchers subsequently started to work on sentiment and opinion and their classifications (Tong 2001; Pang et al. 2002; Das and Chen 2001; Pang et al. 2002). It was Nasukawa and Yi (2003) who first used the term ‘sentiment analysis’ in their research.

The process of sentiment analysis involves detecting three main factors. The first step is to identify whether the given text (sentence) is subjective or objective. Objective sentences do not express any sentiment as they are the factual statements. The next task is to extract the polarity of (subjective) sentences. The final task is detecting the intensity or magnitude of the sentences/document. Sentiment analysis methodology has been essentially designed for accomplishing the above three tasks at different levels, i.e. sentence, aspect and document. The following sections discuss how these tasks have been performed in various studies.

2.1.1 Sentiment Subjectivity

Determining the subjectivity of a word plays a significant role in the sentiment analysis process. Pang and Lee (2004) show that removing objective sentences before the sentiment analysis process begins gives better performance in terms of polarity detection. Some researchers (Lambov et al. 2010; Raaijmakers and Kraaij 2008) have focused on subjectivity classification as distinct from sentiment classification. Similar to sentiment classification, supervised classifiers (which learn algorithms from training dataset) have been used for subjectivity classification. Different features such as unannotated text (Wiebe and Riloff 2005), linguistic features (Xuan et al. 2012) and n-grams (Raaijmakers and Kraaij 2008) have been used to train the classifiers.

2.1.2 Sentiment Polarity

Polarity detection is the most important task in sentiment analysis. Most previous research (e.g. Turney 2002; Shanahan et al. 2005; Dave et al. 2003; Pang et al. 2002) has focused on binary class sentiment (positive and negative) classification. However, Koppel and Schler (2006) showed the importance of learning neutral examples for sentiment classification. Saif et al. (2016)’s work involved four
levels of classification. In addition to the positive, negative and neutral classes, Saif et al. (2016) also considered mixed sentiments which are mixture of both positive and negative sentiments. Research by (Lee and Grafe, 2010; Pang and Lee, 2005) developed a five-level rating (star rating) for sentiment analysis.

### 2.1.3 Sentiment Magnitude

<table>
<thead>
<tr>
<th>Word</th>
<th>Bing Liu’s Lexicon</th>
<th>AFiNN</th>
<th>Vader Sentiment</th>
<th>SentiWordNet</th>
<th>SenticNet</th>
<th>SO-CAL lexicon</th>
</tr>
</thead>
<tbody>
<tr>
<td>brilliant</td>
<td>+1</td>
<td>+4</td>
<td>+2.6</td>
<td>0.875 (POS)</td>
<td>+0.829</td>
<td>+5</td>
</tr>
<tr>
<td>happy</td>
<td>+1</td>
<td>+3</td>
<td>+2.7</td>
<td>0.5(POS)</td>
<td>+0.298</td>
<td>+4</td>
</tr>
<tr>
<td>good</td>
<td>+1</td>
<td>+3</td>
<td>+1.9</td>
<td>0.75 (POS)</td>
<td>+0.883</td>
<td>+3</td>
</tr>
<tr>
<td>glad</td>
<td>+1</td>
<td>+2</td>
<td>+2.0</td>
<td>0.5 (POS)</td>
<td>+0.413</td>
<td>+2</td>
</tr>
<tr>
<td>incapable</td>
<td>−1</td>
<td>−2</td>
<td>−1.6</td>
<td>0.625 (NEG)</td>
<td>−0.736</td>
<td>−1</td>
</tr>
<tr>
<td>sad</td>
<td>−1</td>
<td>−2</td>
<td>−2.1</td>
<td>0.25(NEG)</td>
<td>−0.306</td>
<td>−2</td>
</tr>
<tr>
<td>bad</td>
<td>−1</td>
<td>−3</td>
<td>−2.5</td>
<td>0.875(NEG)</td>
<td>−0.367</td>
<td>−3</td>
</tr>
<tr>
<td>horrible</td>
<td>−1</td>
<td>−3</td>
<td>−2.5</td>
<td>0.625 (NEG)</td>
<td>−0.939</td>
<td>−5</td>
</tr>
</tbody>
</table>

Table 2.1: Some lexical entries with their semantic orientation according to different lexicons

In addition to sentiment polarity, some approaches (e.g. Thelwall et al. (2010); Pang and Lee (2005)) have been interested in calculating sentiment magnitude in sentiment analysis. Magnitude indicates the strength of the text/document’s sentiment. Overall sentiment magnitude is determined by the semantic orientation (ie a sentiment ‘score’) of each lexical item present in the sentence/document (Turney and Littman, 2003). The sentiment analysis approaches based on semantic orientation use different lexicons (lists of sentiment words/lexical items with their semantic orientation or sentiment scores) to obtain an individual word’s semantic orientation. For instance, Taboada et al. (2011) used a lexicon with a sentiment score range between -5 and +5, whereas Esuli and Sebastiani (2006) give positive and negative sentiment words a score between 0 and 1. A number of resources are publicly available for researchers to use in their work. For example, sentiment words or lexical items with appropriate values can be found in Bing Liu’s opinion lexicon[^1], SentiWordNet[^2], AFINN[^3], Linguistic Inquiry and Word Counts[^4] and SenticNet[^5]. However, the semantic orientation or sentiment value of an individual lexical entry in one resource may be different to its value in another resource. Table 2.1 shows the different semantic scores for particular words in several different

[^1]: https://www.cs.uic.edu/~liub/
[^2]: https://sentiwordnet.isti.cnr.it/
[^3]: http://neuro.imm.dtu.dk/wiki/AFINN
[^4]: http://liwc.wpengine.com/
[^5]: http://sentic.net/downloads/
lexicons. For instance, Bing Liu’s opinion lexicon has a list of sentiment words with their appropriate polarity alone, i.e. ‘positive’ or ‘negative’, but it does not contain the magnitude of the words, whereas SentiWordNet has a list of words with their appropriate polarity (sign) and magnitude (numbers) (Table 2.1). Section 2.2.0.2 discusses building the lexicons; they are used with different algorithms and different methods to calculate overall text sentiment and produce a final sentiment class.

2.2 Sentiment Analysis Methodology: Previous Approaches

![Figure 2.1: Sentiment analysis methodology](image)

Sentiment analysis has been considered as a sentiment classification problem by researchers. A classification tries to divide texts up into ‘classes’, rather than giving them a score on a continuous scale which is a regression problem. Different algorithms and rules have been used for sentiment classification problems, like other classification problems. Selecting features, also known as variables/attributes, which capture hidden relevant information is a primary process in classification problems. The appropriate feature sets are then employed with suitable algorithms in sentiment model construction. Similarly, the sentiment analysis methodology consists of a number of important steps, including the feature selection method, as shown in figure 2.1.
In this section, I first discuss feature selection methods, and the techniques that have been used in this area so far. Then I present different approaches that have been used for modelling sentiment analysis. Two main approaches are used for sentiment analysis: the machine learning approach and the lexicon-based approach. Both methods use different techniques to extract the sentiment orientation of a given text. A third method, the hybrid sentiment analysis approach, has also been used by researchers. Hybrid approaches use techniques which are used in both the machine learning and lexicon-based methods.

### 2.2.0.1 Feature Selection

As described above, the feature selection process, or pre-processing, is the first step of any classification process. The input documents for sentiment analysis are texts. In the first phase of pre-processing, the text is broken down into small units, such as word item or stems. Pre-processing involves text cleansing, which is removal of stop words and white space removal, abbreviation expansion, and stemming. This process is also known as transformation (Haddi et al., 2013; Meyer et al., 2008). Then, the features that might be able to create a pattern for future predictions are selected either manually or automatically. Some sentiment analysis approaches use automatic feature selection methods using training datasets, discussed below, whereas other methods use human annotation. The traditional features that have been used for sentiment analysis are: bag-of-words, n-gram, term frequency weight, part-of-speech, opinion words and negations. In the machine learning approach, the system uses the features and patterns to create a model for filtering appropriate outputs (sentiment labels or classes) from the relevant input text. Lexical-based approaches mostly use opinion words as features.

The bag-of-words model is simple and one of the popular feature selection techniques, used to extract uni-grams or opinion words from a text and create a list of words that represent the text (Paltoglou and Thelwall, 2013). Term Frequency-Inverse Document Frequency (TF-IDF) is a numeric statistic that represents a document in a corpus by a word, and has mostly been used for text classification. The TF-IDF value increases with the number of times a word appears in the text. Li and Liu (2010) showed that the TF-IDF method improved the accuracy of their method. Liu and Yu (2014) showed that feature selection using the TF-IDF method shows better performance even for highly imbalanced classes. O’Keefe and Koprinska (2009) used TF-IDF along with feature presence and feature frequency for the feature selection process. Martineau and Finin (2009) show that an improved version of TF-IDF works better.
Chi-square is another feature selection method that has been used in supervised learning methods. The chi-square measurement computes how many specific features are in a particular sentiment class from training datasets. Hagenau et al. (2013) used chi-square along with bi-nomial separation as feature selection methods for stock price prediction from news articles, similar to (Coussement and Van den Poel, 2008) method.

Mutual information (MI) of a term and a sentiment class measures how much the term is informative of its class. Point Mutual Information is the expected value of MI, which has been used in most of the unsupervised learning methods (Turney, 2002; Rothfels and Tibshirani, 2010; Yu et al., 2013), by expanding seed words. Information gain (IG) is the most commonly used feature selection method in the field of machine learning (Yang and Pedersen, 1997; Lee and Lee, 2006). In sentiment analysis, IG analyses the presence and absence of a feature in a document and calculates the relevance of the feature for a sentiment class using its joint probability (Abbasi et al., 2008; Kummer et al., 2012). Ong et al. (2015) obtained better performance results using sparsity adjusted information gain (an improved feature selection metric). Shahana and Omman (2015) found that the IG feature selection method gave better results for both positive and negative classes when compared to the MI, chi squared and TF-IDF methods.

In addition to the above methods, various other methods have been used for feature selection, such as: KL divergence score, used by Kummer and Savoy (2012); the Gini Index-based feature selection method, proposed by Manek et al. (2017); and meta heuristic algorithms, used by Ahmad et al. (2015). Moreover, many other researchers (Liang and Dai, 2013) have used a combination of more than two feature selection methods to improve sentiment analysis performance. Forman (2003) presented an empirical study on twelve feature selection methods. Agarwal and Mittal (2013) used the IG and Minimum Redundancy Maximum Relevancy feature selection methods to extract features of uni-grams, bi-grams and part-of-speech (POS).

2.2.0.2 Sentiment Lexicon

A sentiment lexicon, or a set of polar words, is one of the most important features in sentiment analysis. In fact, lexicon-based approaches mainly depend on the sentiment lexicon. Previously, sentiment lexicons have been created by a variety of different processes. One approach is building the lexicon manually, where researchers mainly use dictionaries or thesauruses, and utilise synonyms and antonyms to determine the polarity of lexical items (Hu and Liu, 2004; Kamps et al., 2004; Kim and Hovy, 2004b; Tong, 2001; Palanisamy et al., 2013). Some researchers (Taboada
et al., 2011; Kiritchenko et al., 2014) have been interested in creating lexicons with intensity scores. Kiritchenko et al. (2014) used multiple annotators, who were asked to rank four different words according to their strength of positivity. They used the annotators’ responses to rank all the words in order. Then the words were scored depending on how far apart from each other they are. POS tags are usually used for creating lexicons manually. Additionally, some other researchers use ‘stemming’ (Palanisamy et al., 2013), ‘exaggerated word shortening’ (Kouloumpis et al., 2011) and ‘emotion detection’ (Ritter et al., 2012) to create lexicons manually.

Other methods involve building the lexicon automatically, and are based on exploiting raw corpus data (Turney, 2002; Turney and Littman, 2003; Hatzivassiloglou and McKeown, 1997b; Esuli and Sebastiani, 2005; Kanayama and Nasukawa, 2006). This approach detects polar sentiment words based on the strength of co-occurrence with polar seed words, which have already been annotated. Kaji and Kitsuregawa (2007) built a Japanese lexicon with the help of polar sentences which had been collected from HTML documents. They used structural clues to extract the sentiment phrases/words. Neviarouskaya et al. (2009) generated a scored sentiment lexicon called SentiFul automatically by using POS words from Neviarouskaya et al. (2007) database. They transformed the intensity of Neviarouskaya et al. (2007) emotional vectors to assign negative and positive scores.

Recently, Labille et al. (2017) created a domain-specific scored sentiment lexicon using probabilities and information theory techniques. They used Amazon product reviews as a domain and used text mining to generate the lexicon, without prior knowledge, and used probabilities and information theory techniques to score the sentiment words.

### 2.2.1 Machine Learning Approaches

In the machine learning approach, the sentiment content of a text unit is identified, extracted or characterised using a classifier. To do this, the sentiments or opinions of the text need to be grouped into categories, such as positive and negative, or positive, negative and neutral, or they can be categorised into an n-point scale, such as strongly positive, positive, weakly positive, neutral, weakly negative, negative and strongly negative or scale-1 to scale-5 (Aly, 2005; Lee and Grafe, 2010; Pang and Lee, 2005). In respect of the statistical approach, a sentiment analysis task can be explained as a classification task, where each category represents a sentiment using relevant features. The primary object of the machine learning approach is to produce an efficient classifier, built using relevant features and suitable algorithms.
A number of machine learning methods have been employed in sentiment analysis in recent years. Either supervised, unsupervised or semi-supervised machine learning methods are used to build a statistical classifier model.

### 2.2.1.1 Supervised Machine Learning

![Supervised machine learning approach to sentiment analysis](image)

Figure 2.2: Supervised machine learning approach to sentiment analysis

The fundamental prerequisite for using the supervised learning method is the availability of labelled text corpora (training datasets) such as the TREC blog tracks\(^6\) (Ounis et al., 2008, 2006) or the MPQA\(^7\) corpus (Wiebe et al., 2005). In the supervised learning approach there are two phases in the process. The first phase is the training phase, where the classifier is built using suitable algorithms and features that are extracted from the training dataset. The second phase is the inference phase, where the trained classifier classifies the sentiments of an unseen (unlabelled) dataset, as shown in figure 2.2.

Supervised learning classifier models are built by different types of algorithm that have a set of input documents, with the desired output or targeted sentiment labels (such as positive, negative, etc.). The algorithms of the classifiers generate a function that maps inputs to desired outputs using features of the input text and output labels. The model is trained until it achieves the desired level of accuracy on the training data. Then the classifier is able to produce output for any unseen input text, according to the algorithms. Various machine learning techniques have modelled different classifiers such as probabilistic, linear, rule-based and decision tree [Medhat](http://ir.dcs.gla.ac.uk/wiki/TREC-BLOG/) [Medhat](https://mpqa.cs.pitt.edu/corpora/)
et al., 2014). Naive Bayes, Maximum Entropy and Support Vector are popular machine learning algorithms have been used by many researchers in past studies (Pang et al., 2002; Bai, 2011; ZHAO et al., 2010; Lane et al., 2012; Ficamos et al., 2017; Gaikwad and Joshi, 2016). Table 2.2 shows some previous supervised learning methods.

The Naive Bayes method builds a simple probabilistic model, which works well on text classification techniques such as topic classification. It uses Bayes’ theorem, based on an independence assumption. In simple terms, Naive Bayes classifiers assume a particular feature of a given class is independent of any other features of a class variable. For example, when a classifier is trained with certain features for positive words, in testing data, the classifier classifies text into positive words based on those features, regardless of the occurrence of other features. Narayanan et al. (2013) show that an enhanced Naive Bayes classifier improves the accuracy of sentiment analysis models, by selecting relevant features. However, the independence assumption is not valid for sentiment analysis, which makes this classifier more confident of its results than it should be.

Maximum Entropy is another probabilistic classifier. However, unlike the Naive Bayes classifier, the Maximum Entropy classifier does not make any independence assumptions. Maximum Entropy is a probability distribution model $p(a, b)$ for class $a$ and context $b$. Training a maximum entropy classifier is a task which involves estimating the probability of class ($a$) being positive or negative with context words ($b$) (Ratnaparkhi, 1997). An efficient Maximum Entropy classifier should have a correct distribution $p(a, b)$ that maximises ‘uncertainty’ or entropy, subject to the constraints and according to the evidence. For example, for a simple Maximum Entropy algorithm, consider it has been told that 40% of documents with the word ‘good’ are in the positive class. If a new document with the word ‘good’ is then tested, the algorithm decides that it has a 40% chance of being a positive document, and a 30% chance each of being a negative and a neutral document (Nigam et al., 1999; Batista and Ribeiro, 2013).

Compared to the Naive Bayes classifier, support vector machines (SVM) are very efficient at traditional text mining, as has been shown by (Joachims, 1998). SVM is a linear model classifier that treats each document as a vector of features in a high-dimensional space. The basic idea of the SVM algorithm is finding out a hyper plane $w$, which separates the document vectors with a largest possible margin in one class from others. Chen and Tseng (2011) used an SVM-based classifier with an information quality framework to evaluate the quality of information in product reviews. Li and Li (2013) proposed a framework based on the SVM algorithm to summarise trending topics and opinions from micro-blogs. Recently, another SVM
classifier was used for aspect-based sentiment analysis by Manek et al. (2017), and showed better performance with the Gini index-based feature selection method.

The decision tree classifier model uses decision tree learning to predict the sentiment class of selected features by learning decision rules inferred from a labelled training dataset. This classifier model uses tree representation to provide a hierarchical composition of the training dataset (Quinlan, 1986). The inner nodes of the tree represent feature values, and each leaf node corresponds to a sentiment label. Recently, Kanavos et al. (2017) explored the decision tree algorithm along with the Naive Bayes algorithm for sentiment analysis in large-scale implementation. However, they showed that the Naive Bayes algorithm worked better. ID3, C4.5, CART and J48 are the popular algorithms used in decision tree learning.

The typical neural network (NN) consists of units (neurons) that are organised in layers, which transform their inputs to an output by applying a function. Then the output is passed on to the next layer. In the training phase in supervised learning, the function is tuned using a set of weights that are associated with each neuron. This is repeated in a multi-layer neural network in multiple layers. In the inference phase, unlabelled inputs are transformed to labelled outputs using the tuned functions in neurons (Chintala, 2012; Duncan and Zhang, 2015). A convolutional neural network (CNN) is a deep, feed-forward neural network, which uses a variation of multilayer perceptions. The major advantage of using CNN in sentiment analysis is that it learns a more general representation by automating the feature generation phase (Stojanovski et al., 2015; Dos Santos and Gatti, 2014; Ouyang et al., 2015). Yin et al. (2017) propose a method that learns sentiment embedding for each word based on sentiment-lexical resources, which are then fed into a CNN classifier. Other deep neutral networks, such as recurrent neural networks (Timmaraju and Khanna, 2015) and recursive neural networks (Socher et al., 2013; Yuan and Zhou, 2015), have also been exploited for sentiment analysis.

Rule-based techniques use a set of rules or manipulation of knowledge to produce a result. The rules are of the form ‘IF some condition THEN some action’. There are a number of criteria involved in generating the rules, which can be generated manually or automatically. For instance, in supervised machine learning approaches, the system learns the criteria and constructs a set of rules automatically. Then it employs those rules for the classifications. These techniques have also been used in the lexicon-based approach, where the set of rules builds the system. I describe the sentiment analysis approach in the following sections. Yang and Shih (2012) proposed a rule mining algorithm model to train effective rules to automatically extract features and sentiment from consumer reviews. Im Tan et al. (2015) built a sentiment analysis system for financial news using rule-based algorithms with a polarity lexi-
con. However, the system did not work well on complex and ambiguous sentences. Chikersal et al. (2015a) combined SVM and rule-based classifiers, and developed a system called SeNTU which produced an average result. Prabowo and Thelwall (2009) Siddiqua et al. (2016b,a) also showed that a rule-based algorithm could be combined with other supervised learning classifiers to produce better results.

In addition to the above classifiers, there are more supervised algorithms such as K-Nearest Neighbourhood (KNN), linear regression (Ginosar and Steinitz 2012) and Random Forest (Parmar et al. 2014), which are used in constructing classifiers for sentiment analysis. Choosing the best classification algorithm is one of the principal steps in supervised machine learning. Many researchers (e.g. Pang et al. 2002; Go et al. 2009; Dey et al. 2016) use more than one algorithm to build the classifier, in order to generate optimal performance. Table 2.2 provides a summary of some previous sentiment analysis studies.
<table>
<thead>
<tr>
<th>Research</th>
<th>Features</th>
<th>Classification Techniques</th>
<th>Text Granularity</th>
<th>Datasets</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pang et al. (2002)</td>
<td>Unigrams, bigrams, POS, adjectives</td>
<td>Naive Bayes, Maximum Entropy, SVM</td>
<td>Document</td>
<td>Movie database (IMDB)</td>
<td>Acc 82.9</td>
</tr>
<tr>
<td>Kennedy and Inkpen (2006)</td>
<td>Adj unigrams, valency shifters</td>
<td>Term counting method using SVM</td>
<td>Document</td>
<td>Movie reviews</td>
<td>Acc 86.2</td>
</tr>
<tr>
<td>Tan and Zhang (2008)</td>
<td>POS, words</td>
<td>Centroid, KNN, Winnow, NB, SVM</td>
<td>Document</td>
<td>Chinese corpus (Education, Movie, House)</td>
<td>F-Score 0.80-0.86</td>
</tr>
<tr>
<td>Go et al. (2009)</td>
<td>Unigram, bigram, POS</td>
<td>Naive Bayes, Maximum Entropy, SVM</td>
<td>Sentence/document</td>
<td>Twitter set</td>
<td>Acc 83.0</td>
</tr>
<tr>
<td>Yessenalina et al. (2010)</td>
<td>Bag-of words, subjective sentence</td>
<td>Structural SVM</td>
<td>Document</td>
<td>Movie reviews, US debate</td>
<td>Acc 88.56-93.22, 70.00 - 77.67</td>
</tr>
<tr>
<td>Lee and Grafe (2010)</td>
<td>n-grams, types of sentences</td>
<td>SVM</td>
<td>Aspect</td>
<td>Restaurant reviews</td>
<td>Acc 57.42</td>
</tr>
<tr>
<td>Kang et al. (2012)</td>
<td>Unigrams, bigrams</td>
<td>Naive Bayes, SVM</td>
<td>Aspect</td>
<td>Restaurant reviews</td>
<td>Maxi P and R 0.78- 0.85</td>
</tr>
<tr>
<td>Authors</td>
<td>Features</td>
<td>Model(s)</td>
<td>Category</td>
<td>Dataset</td>
<td>Score(s)</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------</td>
<td>-------------------------</td>
<td>----------------</td>
<td>------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Mohammad et al. (2013)</td>
<td>Word, character ngrams, lexicons, POS, all-caps</td>
<td>SVM</td>
<td>Message, term</td>
<td>Tweets, SMS</td>
<td>F-score</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Message- 69.02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Term-88.93</td>
</tr>
<tr>
<td>Chikersal et al. (2015b)</td>
<td>n-grams, linguistics characters</td>
<td>SVM</td>
<td>Sentence</td>
<td>Twitter sets</td>
<td>F-score</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>81.90</td>
</tr>
<tr>
<td>Moh et al. (2015)</td>
<td>n-gram, POS</td>
<td>Naïve Bayes, SVM, Random Forest, SGD</td>
<td>Document</td>
<td>Movie review data from Rotten Tomatoes database</td>
<td>Acc</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>80.53- 87.23</td>
</tr>
<tr>
<td>Zimbra et al. (2016)</td>
<td>Unigram, ngram, valency, shifters</td>
<td>Neural network (DAN2)</td>
<td>Tweet/ messages</td>
<td>Starbucks tweets</td>
<td>Acc</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3-class- 86.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5-class-85.56</td>
</tr>
<tr>
<td>Yang et al. (2016)</td>
<td>Words and phrases</td>
<td>(Treebank convolutional) neural network</td>
<td>Sentence</td>
<td>Movie reviews</td>
<td>Acc</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Binary- 95.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Fine grained-49.99</td>
</tr>
<tr>
<td>Liu et al. (2017)</td>
<td>Text features</td>
<td>Decision tree, NB, SVM, radial basis function neural network, KNN</td>
<td>Document</td>
<td>Chinese reviews (12 dataset)</td>
<td>Acc</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>best-82.50</td>
</tr>
</tbody>
</table>

Table 2.2: Summary of some previous supervised learning approaches
2.2.1.2 Unsupervised Machine learning

Supervised learning approaches build upon a set of fully annotated data, which is used to train a classifier via certain algorithms. However, these supervised learning methods require labelled datasets, which are usually very expensive to obtain. Moreover, their availability is limited. On the other hand, unsupervised learning approaches do not use training data. This method employs unsupervised algorithms to infer a function automatically, in order to represent the hidden structure of unlabelled data. The main idea of this approach involves using ‘seed words’ to automatically build a lexicon or knowledge base, which is used to classify newly incoming text. Some unsupervised learning methods populate sentiment seed words via clustering and association methods, using different algorithms such as the k-means, TF-IDF and PMI-IR algorithms \cite{Turney2002, Zagibalov2008, Unnisa2016}.

Point Mutual Information (PMI) measures the association between two words \cite{Church1990, Turney2002}. Turney \cite{Turney2002} used PMI with information retrieval (IR) to calculated the semantic orientation of a document by using the PMI of a phrase and gold standard reference positive \textit{(excellent)} and negative \textit{(poor)} words by using the similarity of the phrase and the reference seed word; thus the final output was the aggregated semantic orientation of the words in the document.

The K means algorithm supports the extraction of insights from a large corpus by grouping in clusters based on similar observations of units \cite{Hartigan1979}. The first step of the algorithm is to form K word seeds, and to partition other words into K clusters based on a distance function \cite{Li2010}.

Other unsupervised machine learning algorithms, such as Latent Dirichlet Allocation (LDA) and the Hidden Markov Model (HMM) have been exploited for sen-
Semi-supervised learning is another method of machine learning, which falls between supervised and unsupervised learning. Semi-supervised learning algorithms make use of labelled and unlabelled data to model classifiers (Zhu, 2005). Various algorithms have been used to construct the classifier in different ways. For example, Yang et al. (2015) utilized lexicon-based learning and corpus-based learning to train a classifier.
<table>
<thead>
<tr>
<th>Research</th>
<th>Features</th>
<th>Classification Techniques</th>
<th>Text Granularity</th>
<th>Datasets</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turney (2002)</td>
<td>POS, phrases</td>
<td>PMI-IR</td>
<td>Document</td>
<td>Epinions, reviews of automobiles, banks, movies and travel</td>
<td>Acc 68-84</td>
</tr>
<tr>
<td>Dasgupta and Ng (2009)</td>
<td>Vector of unigram</td>
<td>Spectral clustering</td>
<td>Document</td>
<td>Movies, Amazon reviews</td>
<td>Acc 77.6-99.8</td>
</tr>
<tr>
<td>Lin and He (2009)</td>
<td>Paradigm words</td>
<td>LDA</td>
<td>Document</td>
<td>Movie reviews</td>
<td>Acc 84.6</td>
</tr>
<tr>
<td>Rothfels and Tibshirani (2010)</td>
<td>Seed words</td>
<td>PMI-IR and iterative algorithms</td>
<td>Document</td>
<td>Movie reviews</td>
<td>Acc 65.5</td>
</tr>
<tr>
<td>Brody and Elhadad (2010)</td>
<td>Adjectives, negation</td>
<td>LDA for aspect detection; propagation method</td>
<td>Document</td>
<td>Restaurant reviews</td>
<td>Acc n/a</td>
</tr>
<tr>
<td>Ortega et al. (2013)</td>
<td>Emoticons, slang term, POS, lemma</td>
<td>Rule-based</td>
<td>Sentence</td>
<td>Tweets and SMS</td>
<td>F-score 50.17/44.39</td>
</tr>
<tr>
<td><strong>Hu et al. (2013)</strong></td>
<td>Emoticons, words</td>
<td>Tri-matrix factorization</td>
<td>Sentence</td>
<td>Tweets and Obama-McCain debate</td>
<td>Acc. 0.726/0.692</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------</td>
<td>-------------------------</td>
<td>----------</td>
<td>-------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><strong>Chifu et al. (2015)</strong></td>
<td>Words</td>
<td>Growing hierarchical self-organising maps</td>
<td>Aspect</td>
<td>Customer reviews</td>
<td>Acc 63.15</td>
</tr>
<tr>
<td><strong>Unnisa et al. (2016a)</strong></td>
<td>Uni-gram, bi-gram</td>
<td>Spectral, K-means, hierarchical clustering</td>
<td>Sentence</td>
<td>Tweets</td>
<td>Acc 88.56-93.22</td>
</tr>
<tr>
<td><strong>Suresh and S. (2016)</strong></td>
<td>Words</td>
<td>Fuzzy clustering algorithm</td>
<td>Sentence</td>
<td>Tweets about Samsung Galaxy</td>
<td>Acc 76.4</td>
</tr>
<tr>
<td><strong>Vilares et al. (2017)</strong></td>
<td>Lexicon</td>
<td>Rule-based (syntax)</td>
<td>Sentence</td>
<td>Previous corpora (movies)</td>
<td>Acc 74.25</td>
</tr>
<tr>
<td><strong>Lo et al. (2017)</strong></td>
<td>Term</td>
<td>K means, Twitter LDA, DPMM clustering</td>
<td>Sentence</td>
<td>Tweets</td>
<td>Acc xx</td>
</tr>
</tbody>
</table>

Table 2.3: Summary of some previous unsupervised learning approaches
2.2.2 Lexicon-Based Sentiment Analysis

The lexicon-based approach is an unsupervised method, and mainly relies on sentiment lexicons. This approach predominantly uses opinion words and POS tags as features. In the early years, the basic lexicon-based method involved counting the number of positive and negative words in sentences/documents (Pennebaker et al., 2007). If the number of positive words was more than negative words, then the text was considered a positive sentiment, otherwise, negative. Turney and Littman (2003) introduced the idea that the semantic orientation of a word indicates its sentiment. Accordingly, lexicon-based methods for sentiment analysis started to utilise lexical items and their semantic orientation or sentiment scores (Ding et al. 2008, Hu and Liu 2004), and used various lexical resources that contain a list of sentiment words that are assigned with semantic/sentiment scores (such as Sentiwordnet (Esuli and Sebastiani 2006) and Q-wordnet (Agerri and García-Serrano 2010)). Building lexicon has already been discussed in section 2.2.0.2. Some researchers (e.g. (Taboada et al., 2011; Palanisamy et al., 2013)) created the sentiment lexicons for their lexicon-based approaches, whereas others (e.g. (Rajput et al., 2016)) used publicly available lexicons such as MPQA\footnote{http://mpqa.cs.pitt.edu/lexicons/}.

![Figure 2.4: Lexicon-based approach methodology](image-url)

The approach aggregates the semantic orientation score of the lexical items in a given text to obtain a final sentiment score for the text. However, due to the com-
plexity of linguistic features, only aggregating scores of the text has not been suffi-
cient to produce effective results. Polanyi and Zaenen (2006); Kennedy and Inkpen
(2006)'s studies explored how the sentiment of lexical items is modified by valence
shifters such as negation and intensification. Additionally, some researchers (Ding
et al., 2008) exploited connectives and conjunctions to handle contextual informa-
tion, while others such as Taboada et al. (2011) added relevant rules and algorithms
for handling valence shifters, such as intensification, negation and irrealis blocking
in the final calculation of the sentiment score of the text. These valence shifters are
not sentiment words but they change the semantic orientation of their neighbouring
words. Figure 2.4 shows the basic methodology of the lexicon-based approach, and
table 2.4 provides a summary of some previous lexicon-based approaches.
<table>
<thead>
<tr>
<th>Research</th>
<th>Lexical Resources</th>
<th>Rules and Algorithms</th>
<th>Text Granularity</th>
<th>Datasets</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ding et al. (2008)</td>
<td>Manually created in previous study</td>
<td>Negation, orientation equation, linguistic conventions</td>
<td>Aspect</td>
<td>Customer product reviews</td>
<td>F score 0.90</td>
</tr>
<tr>
<td>Taboada et al. (2011)</td>
<td>Manually created</td>
<td>Intensification, negation, irrealis blocking, text features</td>
<td>Document</td>
<td>Epinions, movie reviews</td>
<td>Acc 78.74</td>
</tr>
<tr>
<td>Maks and Vossen (2011)</td>
<td>Dutch Wordnet (DWN and Dutch Reference Lexicon (DRL))</td>
<td>This is a lexical model-detecting subjectivity and sentiment using verbs</td>
<td>Document / sentence</td>
<td>Real-time tweets</td>
<td>Agreement per 0.84-0.92</td>
</tr>
<tr>
<td>Palanisamy et al. (2013)</td>
<td>Manually created</td>
<td>Negation, blind negation, clause split, total aggregation</td>
<td>Document / sentence</td>
<td>Real-time tweets</td>
<td>Precision 0.9361 / 0.8884 Recall 0.7132 / 0.7912</td>
</tr>
<tr>
<td>Ngoc and Yoo (2014)</td>
<td>AFINN</td>
<td>Intensification, negation, calculating sentiment score, satisfaction Score of post, satisfaction score of fan page total</td>
<td>Document</td>
<td>Facebook posts</td>
<td>Acc n/a</td>
</tr>
</tbody>
</table>

https://www.cs.uic.edu/~liub/FBS/sentiment-analysis
http://neuro.imm.dtu.dk/wiki/AFINN
<table>
<thead>
<tr>
<th>Name</th>
<th>Methodology</th>
<th>Data Source</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jurek et al. (2015)</td>
<td>Generated based on SentiWordnet</td>
<td>Document</td>
<td>Acc 63.1 - 77.3</td>
</tr>
<tr>
<td></td>
<td>Negation, intensification, proposed combining function (normalisation)</td>
<td>Stanford Twitter test set and IMDB data set</td>
<td>51.4 - 74.2</td>
</tr>
<tr>
<td>Rajput et al. (2016)</td>
<td>MPQA</td>
<td>Document</td>
<td>Acc 91.2</td>
</tr>
<tr>
<td></td>
<td>Word frequency, words attitudes, overall attitude and sentiment score using summation of overall attitude</td>
<td>Student feedback</td>
<td></td>
</tr>
<tr>
<td>Aung and Myo (2017)</td>
<td>Manually created lexicon and AFINN</td>
<td>Document</td>
<td>Acc n/a</td>
</tr>
<tr>
<td></td>
<td>Negation, blind negation, intensification, total sentiment using heuristic techniques</td>
<td>Student comments</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.4: Summary of some previous lexicon-based approaches for sentiment analysis

11 http://sentiwordnet.isti.cnr.it/
12 http://mpqa.cs.pitt.edu/lexicons/
13 http://neuro.imm.dtu.dk/wiki/AFINN
2.2.3 Hybrid Approaches

Both the lexicon-based and machine learning approaches to sentiment analysis have their strengths and weaknesses. Some researchers have exploited techniques that are used in both methods, in order to obtain the best of both approaches. Different hybrid methods have been implemented in various ways. Maurel et al. (2008) hybrid approach produced final results by taking an average output of both the symbolic and statistical methods. In other hybrid classification processes (e.g. Mudinas et al. (2012)), both classifiers have been set in a sequential order, which allowed one method to exploit the information in the results of another approach. Balage Filho and Pardo (2013), Zhang et al. (2011) have proved that the process works well for Twitter messages. Malandrakis et al. (2013) used a lexicon-based model to train a large external dataset for their supervised machine learning approach. Similarly, Mudinas et al. (2012) used a lexicon-based approach to generate feature vectors for a supervised machine learning approach. Andreevskaia and Bergler (2008) developed a system using a hybrid method for the cross domain sentiment analysis. They trained a statistical classifier on a small dataset and used the classifier within the same domain, and they used lexical approach for the other domains.

2.2.4 Other Previous Research

Another technique, concept-level sentiment analysis, has been becoming popular in recent years (Cambria, 2013; Poria et al., 2014). This technique leverages not only semantic knowledge but also common-sense knowledge. The concept-level approach, along with common-sense knowledge, has been used for detecting sarcasm by Tunghamthiti et al. (2014). Detecting sarcasm has been a challenging task in opinion mining. However, in recent years, many researchers attempted to solve the issue (Maynard and Greenwood, 2014). Contextual information in particular has been useful in detecting sarcasm (Riloff et al., 2013). Similarly, many researchers have started to tackle contextual information for sentiment analysis. For example, Muhammad et al. (2016) introduced a sentiment analysis system which used local and global context by generating a hybrid lexicon. Wilson et al. (2005), Saif et al. (2016) explored contextual semantics for sentiment analysis.

Although types of sentence play a significant role in sentiment analysis, this has not been broadly exploited for sentiment analysis by many researchers. Comparative sentences, for instance, are used in product reviews to express satisfaction or disappointment with a product by comparing it to its competitors’ products (Ganapathibhotla and Liu, 2008; Zhang et al., 2011). Conditional sentences are another
challenging type of sentence that make it hard to determine the exact opinion or sentiment. Narayanan et al. (2009) used linguistic knowledge such as conditional connectives to build a supervised model for sentiment analysis of conditional statements. The impact of language features such as modality and negation have also been widely exploited for sentiment analysis (Benamara et al., 2012; Kiritchenko and Mohammad, 2016).

All the above approaches have conducted all the levels of sentiment analysis tasks, i.e. document-level, aspect-level and sentence-level. General sentiment analysis methodologies are used for document- (Moraes et al., 2013; Yessenalina et al., 2010; Behdenna et al., 2016; Taboada et al., 2011) and sentence-level tasks (ZHao et al., 2010; Yang and Cardie, 2014). An aspect-level task (ABSA; Aspect Based Sentiment Analysis) typically includes several subtasks, which include detecting aspects and aspect terms. Previous approaches have used various techniques to identify target aspects and aspect terms, such as frequent term identification (Hu and Liu, 2004), training a classifier using labelled data (Varghese and Jayasree, 2013) and dependency parsing (Jiang et al., 2011).

The ABSA task has been added to the annual SemEval competition since 2014. SemEval is an ongoing evaluation series in the field of computational semantic analysis. In subsequent years, aspect term extraction, aspect term polarity and aspect category have also been the focus of subtasks of ABSA in SemEval competitions. Entity-level sentiment analysis tasks were also introduced as subtasks of SemEval-2016 (Nakov et al., 2016). Previous years’ SemEval competitions have focused sentiment analysis tasks on a few more areas too. SemEval-2014 (Ghosh et al., 2015) conducted a sentiment analysis task on figurative language, including sarcasm. Another series at SemEval-2015 introduced a task that classifies an event as pleasant or unpleasant (Russo et al., 2015). Stance detection is another sub-task of sentiment analysis, which detects whether the author is in favour of or against a given targeted event/description (Mohammad et al., 2016). This work was included in SemEval-2016 using a set of tweets for the evaluation. The candidate system evaluation results showed that the systems found it difficult to understand stance towards the target of interest from tweets that expressed an opinion towards another entity.

14 http://alt.qcri.org/semeval2014/task4/
15 http://alt.qcri.org/semeval2016/task4/
16 http://alt.qcri.org/semeval2015/task11/
17 http://alt.qcri.org/semeval2015/task9/
18 http://alt.qcri.org/semeval2016/task6/
2.3 Inheritance Models

Inheritance is a mechanism for grouping similar properties in a hierarchical structure. The major advantage of inheritance is to minimise the number of duplicate properties. Fahlman (1979) used hierarchy and inheritance features to build a system called NETL, which represents and uses real world systems. Etherington and Reiter (1983) proposed an extended version of the NETL system by formalising inheritance hierarchies with some exceptions using a default logic. Smolka and Ait-Kaci (1989) used an inheritance hierarchy structure to represent taxonomically organised data, in order to build a framework that accommodates feature types. POETIC, a traffic information collator, was developed to use an inheritance-based lexicon system by Gaizauskas (1992); Evans et al. (1995). Recently, Black et al. (2016) discussed the essence and importance of inheritance in programming languages.

A non-monotonic (or default) inheritance network is a collection of nodes. Each node is associated with specific features or rules and is organised into hierarchies which have been allowed to have exceptions. Fraser and Hudson (1992) discussed the central role played by default inheritance in word grammar. Non-monotonic inheritance networks of lexicons have a long history in natural language processing. Daelemans et al. (1992) showed that a non-monotonic inheritance mechanism was essential for lexical networks. In the same year, Russell et al. (1992) built a unification-based lexicon system for NLP applications using the mechanism of multiple default inheritance.

Several types of lexicon formalisms have been developed using a non-monotonic inheritance-based network, such as DATR (Evans and Gazdar 1996), IBT (Hartrumpf 1994) and LRL (Copestake 1992). Evans (2013) introduced the extended lexicon framework (ELF), a new direction of development in DATR, which can represent words not in isolation, but as instances occurring in sentences.

2.4 Evaluation of Sentiment Analysis

Evaluation is an important process when estimating the performance of text/data classification in information extraction or natural language processing systems (Goutte and Gaussier 2005; Fawcett 2006). The accuracy of a classifier is typically measured based on its precision, recall, f-score and accuracy values. For a simple binary classification task, a set of documents is given to a system to filter, dependent on some given features. Consequently, I obtain two sets of documents. One is filtered (accepted) by the system, and the system rejects the other one. The evaluation of
this task is to identify how good this system is at separating relevant documents from irrelevant documents. Table 2.5 shows a $2 \times 2$ confusion matrix or contingency table according to the classification.

<table>
<thead>
<tr>
<th>Human/gold standard says yes</th>
<th>Machine says yes</th>
<th>Machine says no</th>
</tr>
</thead>
<tbody>
<tr>
<td>$tp$</td>
<td>$fn$</td>
<td></td>
</tr>
<tr>
<td>$fp$</td>
<td>$tn$</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.5: The outcomes of the binary classification formulated in a confusion matrix

In the typical text classification evaluation method:

- true positive/ $tp =$ number of correct documents/ items classified by the system.
- false negative/ $fn =$ number of relevant documents which are rejected by the system.
- false positive/ $fp =$ number of documents that are incorrectly identified as relevant by the system.
- true negative/ $tn =$ number of non-relevant document that are rejected by the system.

The precision of this system is the fraction of filtered documents that are relevant, while recall is the fraction of the relevant document that is filtered. F-score is a single value, which is a combination of precision and recall. The accuracy value is calculated as number of documents correctly classified by the system. The following equations are used for the calculation:

- Precision ($P$) = $\frac{tp}{tp + fp}$
- Recall ($R$) = $\frac{tp}{tp + fn}$
- Accuracy ($A$) = $\frac{tp + tn}{tp + tn + fp + fn}$
- F-Score ($F$) = $\frac{2PR}{P + R}$

Since sentiment analysis is a branch of text classification (Jurafsky and Martin, 2015), the evaluation method for text classification is also adopted for sentiment analysis evaluation, and precision, recall and f-score values are used to measure the performance of the system (Turney, 2002; Pang et al., 2002; Nasukawa and Yi, 2003; Prabowo and Thelwall, 2009; Turney, 2002). Unlike standard text classification, sentiment classification involves more than two classes. Sentiment analysis studies focus
on three categories: positive, negative and neutral. However, some research suggests having an extra class, adding to the neutral class the mixed-sentiment class, which is a mixture of positive and negative opinions (Saif et al., 2016). Pang and Lee (2005) and Nakov et al. (2016) are interested in 4- or 5-star scales/classifications. For these types of multi-classification task, precision, recall and f-score values (performance measures) are calculated for each class. Then, the performance measures for the whole system are calculated by averaging those values using micro- or macro-averaging (Prabowo and Thelwall, 2009).

Multi-class sentiment classification has been considered to be ordered classification. In the evaluation process, researchers have also started to focus on how far the system’s misclassifications deviate from the actual class. For example, consider a 5-class sentiment classification system (scale 1 to scale 5). Classifying a scale-5 document as scale-1 class is a worse error than classifying it as a scale-4 class. To measure these types of error, an additional evaluation measure, macro-averaged mean absolute error (MAE$^M$) was used in SemEval 2016 by Nakov et al. (2016).

Macro-averaged mean absolute error is calculated as follows:

$$\text{MAE}^M = \frac{1}{|c|} \sum_{j=1}^{|c|} \frac{1}{N_j} \sum_{i=1}^{N_j} |a_i - p_i|$$

where, $c$ is number of classes; $N_j$ is number of documents of $j^{th}$ class; $a_i$ is actual sentiment label (classes) of $i^{th}$ document; $p_i$ is predicted sentiment label (class) of $i^{th}$ document; and $|a_i - p_i|$ is distance between actual and predicted sentiment classes of $i^{th}$ document.

In addition, quantification measures, Kullback-Leibler Divergence (KLD) for binary classification and Earth Mover’s Distance (EMD) for multi-class classification for evaluation, were introduced in SemEval-2016-task4 (Nakov et al., 2016). They used quantification for estimating the prevalence of sentiment tweets about a given topic. Similarly, in SemEval-2015-task11 (Ghosh et al., 2015) and SemEval-2017-task5, cosine similarity was used to assess the comparison of sentiment predictions of each participating system and the human-annotated gold standard for a set of tweets. These evaluation measures are relevant to the evaluation of entity-level sentiment analysis task.

---

2.5 Research Gaps

Various approaches have been applied in the field of sentiment analysis. Although there has been an increased focus on sentiment analysis in recent years, there are still several research gaps, and a lack of explanations within the area, and I have identified and addressed these in this thesis.

- Machine learning methods work well only in bounded domains. Moreover, they require larger training datasets. On the other hand, lexicon-based approaches rely on lexicons and need expert manual contractions. Many researchers have already discussed these issues and proposed different hybrid methods. However, the effect of linguistic context has not been adequately exploited. Moreover, the vast varieties of techniques and rules have not yet been implemented in a single framework. I aimed to address this issue by implementing various rules and algorithms in a single framework using an inheritance-based mechanism.

- Context-dependent opinion words, such as descriptive adjectives (e.g. big, small, long, short), have been handled with the help of external reviews using linguistic conventions (Ding et al., 2008) or pattern-based methods (Wu and Wen, 2010) using training datasets. However, the sentiment of the descriptive adjectives of one review is not necessarily the same as only in the previous clause, external reviews or training datasets. Thus context-dependent opinion words are can be widely explored within the same author’s view in the same review. This thesis focused on tackling this problem by considering contextual information for the descriptive adjectives.

- Koppel and Schler (2006) showed that learning neutral examples contributes to the accuracy of the positive and negative classes of sentiment analysis. However, a definition of a neutral class has not so far been precisely defined. In this thesis, I defined the neutral class by introducing a sentiment neutral model.

- Another main gap is the lack of a suitable sentiment framework that calculates sentiment score accurately at the sentence level, as stated as challenge (1) in section 1.4. This project develops a framework to overcome this gap.

- Supervised and unsupervised machine learning methods produce labelled outputs without any direct interpretation of what the classes ‘mean’, using training datasets and various algorithms. The lexicon-based methods proceed by aggregating semantic orientation (a numerical score obtained from various lexicon sources) and deciding the sentiment of the document, depending on its
sign and magnitude. The aggregation operations involved also vary, and do not always have straightforward semantic interpretations. Comparing the outputs of those systems, or bringing those outputs into a common scale, is therefore very challenging. This thesis introduces a calibration method which maps numerical scores onto classes. This also overcomes challenge (2) as stated in section 1.4.

This project also provides a framework that could be extended to overcome some of challenges stated in section 1.4. For instance, the contextual polarity feature of my sentiment framework could tackle sarcasm (challenge 6) in section 1.4. Moreover, a lexical item within my sentiment framework can access its neighbouring lexical items. This feature could be extended to handle anaphora resolution (challenge 3).

2.6 Research Methodology

This research is an extension of existing lexical-based approaches. I first set out to study two existing lexicon-based approaches used for sentiment analysis and to replicate their key analysis algorithms as ELF rules. I selected Taboada et al. (2011)’s ‘Lexicon-Based Methods for Sentiment Analysis’ (the SO-CAL system) and Ding et al. (2008)’s ‘A Holistic Lexicon-Based Approach to Opinion Mining’ (the Opinion Observer system). Then I aimed to model them in Galadriel, using as far as possible the same datasets and the features which are used in those existing systems. I evaluated Galadriel against both original works, so that I could demonstrate the principle that sentiment knowledge can be modelled in the DATR/ELF inheritance framework. Then I merged both Liu et al.’s Galadriel model and Taboada et al.’s Galadriel model, while identifying novel techniques. From these analyses, an integrated inheritance model of sentiment knowledge of words is identified and extended to a model of sentiment analysis. In this way, the entire sentiment analysis task can be encoded as a ‘lexical description’ task.

The final step is to introduce insights from other research approaches; in particular, corpus pattern analysis techniques are used to populate lexicons from examples and added to the model. To illustrate, I want to use Galadriel to handle phrases that are commonly used in web documents and reviews. In order to handle such phrases, a model has been added into Galadriel, using a corpus-based analysis methodology to refine this model with examples derived from corpus data. I only exploit the idea behind the corpus-based techniques to tackle irregular lexicon items or small phrases that are not commonly present in sentiment dictionaries. I do not attempt to use automatic acquisition using machine learning techniques. These models are
evaluated by comparing them with the existing methods.

Finally, the *Galadriel* system was evaluated using the quantitative approach in different level tasks, i.e. sentence-level, document-level and aspect-level. I used various datasets for the different levels of analysis. As discussed above, evaluation is usually done by comparing the results of the sentiment system with gold standard results, and computing precision, recall and f-score values. I used the same performance measures to compare the *Galadriel* system with the gold standard results. *Galadriel* views sentiment analysis as a regression task, but the gold standard systems view it as a classification task. I introduce a calibration method which optimises the mapping from regression output to classes.

## 2.7 Summary of the Chapter

This chapter has presented a review of the literature on sentiment analysis. I started with the definition of sentiment analysis. I then reviewed previous research methodologies. Then I discussed the approaches and evaluation methods that have been conducted by various researchers in past studies. I also discussed how the inheritance structure has been exploited in NLP. In addition, I have outlined gaps in previous research in the field of sentiment analysis. Finally, I outlined my research methodology.
Chapter 3

Inheritance-Based Lexical Knowledge Representation

Inheritance networks consist of nodes and associations between them in a hierarchical structure. Every node has some information that is shared by its sub-nodes. In non-monotonic, or default, inheritance networks, information is usually inherited by lower nodes of the hierarchy; however, exceptions can override this inheritance if required. This thesis aimed to exploit the application of a non-monotonic inheritance-based lexicon for sentiment analysis. To do this, I used DATR (Evans and Gazdar 1996), a knowledge representation language that defines non-monotonic inheritance networks to describe lexical information. This chapter introduces DATR and a recent extension ELF (the extended lexicon framework, Evans 2013), and outlines the lexical modelling approach used in this thesis.

3.1 Overview of DATR and ELF

Lexicons are fundamental components of a language and play a vital role in linguistic theory and natural language processing systems. Lexical knowledge represents information of lexical items and their relationships, including linguistics knowledge but also conceptual and parametric knowledge. The construction of lexicons requires lexical rules and knowledge, which has led many linguists and NLP researchers to adopt a theory of lexical representation languages. In some of these representation languages, inheritance structures and rule bases have been used for encoding of lexical information, in a way which is similar to that seen in modern ontology languages (such as OWL) and object-oriented programming languages (Java, C++, Python, etc.). However, these object-oriented programming languages are not designed for
representing linguistic knowledge and are not ideal for context-specific NLP tasks such as the lexical approach to sentiment analysis. Hence, lexical knowledge representation languages were developed based on formal models of lexical knowledge such as unification-based attribute-value matrices (AVM), directed acyclic graphs (DAG).

In the mid 1980s, when the development of unification-based grammar formalisms was at its height, but lexical knowledge was still quite primitively represented, Evans and Gazdar began to explore some interesting technical questions about how lexicons could be organised in a structure that could capture both subregularity and irregularity while minimizing redundancy. They found that this was the key theoretical issue and aimed to tackle it by using semantic nets in the KL ONE tradition (Brachman, 1979) as well as lexical rules. As a result, Evans and Gazdar (1996) developed a language for lexical knowledge representation called DATR, where lexical knowledge is encoded in a network of nodes. DATR is broadly based on PATR (Shieber et al., 1983), one of the earliest ‘programming languages’ for linguists, based around unification of AVMs. Although it is a powerful grammar-writing system, PATR does not seek to represent any particular linguistic theory or approach. DATR shares this property, providing a language for describing lexical phenomena without being theoretically prescriptive. However, descriptively, DATR is more powerful than PATR, in particular because it makes use of default inheritance of information between the linguistic concepts it represents.

DATR itself is quite old, but by way of comparison, the main lexical knowledge representation languages such as LRL (Copestake, 1992) and IBL (Hartrumpf, 1994) are similarly quite old. IBL has not been used for any other systems. However, LRL is still in active use in the Stanford HPSG system (Copestake and Flickinger, 2000). Similar to LRL and IBL, DATR uses an inheritance mechanism. However, DATR makes more use of defaults than either of the other two. The DATR lexicon structure supports more granularity of lexical knowledge. This allows me to define the sentiment lexicon with a higher degree of flexibility than the notion of the superclass in the other inheritance-based lexicons.

From the late 1990s, DATR started to be used in different projects and systems. For example, the DATR lexicon was used for the project GREC[1], which was a project that aimed to develop a multilingual (Georgian, Russian, English and German) valency lexicon for use in various NLP applications. Then, a system called KATR, an extension of DATR specifically designed for modelling inflectional morphology, was developed by Finkel et al. (2002). In 2001, Tiberius (2001) PhD thesis explored different architectures for multilingual lexicons, implementing lexical fragments in

[1]http://www2.informatik.uni-stuttgart.de/ivi/is/greg-index.html
DATR. The application of DATR has also been exploited in different languages, such as Russian (Evans et al., 2003) and Bulgarian (Stoykova, 2010).

ELF is a more recent addition to DATR, which extends the language from a lexical framework (representing individual words) to a grammar framework (representing sentences). Grammar frameworks have been used for writing grammars of natural languages. Context free grammar (CFG) is a popular framework, and is an entirely rule-based system without lexicalisation. Later, Gazdar et al. (1985) showed that syntax can be described by CFG with suitable conventions, and introduced the GPSG (generalized phrase structure grammar) framework with more powerful rules, but including only a very small amount of lexicalisation. On the other hand, HPSG (Pollard and Sag, 1994) and LTAG (Joshi and Schabes, 1991) are highly lexicalised grammar formalisms. Similar to ELF, lexical entries are structured in a hierarchy based on types in HPSG and involving only a few rules. Similarly, LTAG creates a tree-based lexicon using a combination of rules. However, ELF is more lexically oriented compared to the HPSG and LTAG frameworks. Later still, Sign-Based Construction Grammar (SBCG), which is a constructional version of HPSG, was introduced by Fillmore et al. (2007). The sign is an important type of feature in SBCG and word, phrase and lexemes are its subtypes. There is no core grammar in SBCG and it is a licensing-based theory. ELF itself is theory-neutral, like DATR and PATR.

Grammar frameworks use grammar rules and lexicons and have been used for sentiment classification for many years. For instance, the CFG framework has been used to formulate sentiment grammars for statistical parsing (Dong et al., 2015) and syntactic rule-based sentiment analysis (Mavljutov and Ostapuk, 2013). The HPSG lexical definition has been exploited for parsing in sentiment analysis (Ben-Ami et al., 2014).

### 3.2 DATR: A Language for Lexical Knowledge Representation

Evans and Gazdar (1996) designed DATR, a lexical description language, to model the structure of a lexicon using default inheritance. The core descriptive unit in DATR is a node. Each node has a unique name, and associated with each node is a set of definitional path equations mapping paths (sequences of features) onto value definitions. Evans and Gazdar introduce the basic ideas of DATR using the following simple example: consider two verbs, love and walk, with some of their morphological forms:
Figure 3.1: The abstract node VERB is defined by DATR. Source: Evans and Gazdar (1996)

The morphological forms of both verbs (love, walk) have the same pattern. For example, the present participle of love and walk is the root word with ing attached, which are called extensional statements. Similarly, the morphological forms of most other verbs have the same patterns. This information is shared between all verbs in order to avoid duplication. Evans and Gazdar organised this information as a network of nodes in DATR, where a node is a collection of the information...
that has been shared. A node is similar to a word, lexeme or a class of lexemes. Each node of the network is associated with a set of **path/feature:value** pair statements that could capture generalizations. They name these statements **definitional** statements. For example, the VERB node is associated with the following set of path/feature:value pairs.

<table>
<thead>
<tr>
<th>path/feature</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>syn cat</td>
<td>verb</td>
</tr>
<tr>
<td>mor</td>
<td>&quot;&lt;root&gt;&quot;</td>
</tr>
<tr>
<td>mor pres 3sing</td>
<td>&quot;&lt;root&gt;&quot; s</td>
</tr>
<tr>
<td>mor pres part</td>
<td>&quot;&lt;root&gt;&quot; ing</td>
</tr>
<tr>
<td>mor past part</td>
<td>&quot;&lt;root&gt;&quot; ed</td>
</tr>
</tbody>
</table>

The above shows that the VERB node’s syntactic category is the verb, and its morphology is substituted with the root word. Similarly, its morphological forms for present third-person singular, present participle and past participle are the root word with s, *ing* and *ed* attached, respectively. These definitions are accessed by looking up a path/feature in the node and returning the corresponding value. If a path is requested which is not defined in the node, the definition with the longest leading subpath is used instead. So for all the morphological forms apart from the three explicitly listed, the shorter path <mor> will be used. In other words, by default, the morphological form of a verb is its root, apart from the three listed cases. This definitional statement, coloured purple, is written in DATR as shown in figure 3.1. The above information passes to its sub class, the LOVE node, and sets the <root> feature to *love* (VERB does not define it at all - abstract verbs have no forms), which is written in DATR as:

```
LOVE :
< >  ==  VERB
<root>  ==  love
```

Notice the ‘empty’ path definition < > == VERB. The empty path will always match a looked-up path, if nothing else does. So this is the ultimate default definition. In other words, this line says ‘if a definition is not provided here, inherit it from VERB’. In figure 3.1, this is represented by the inheritance line between the nodes.

Moreover, figure 3.1 shows the extensional statements (results, rather than definitions) that can be derived in DATR, as shown by the red colour. In DATR, angle brackets (< .... >) determine the boundaries of paths, and the equality operator is used to differentiate the statements: = is used for **extensional** statements, while == is used for **definitional** statements.
Furthermore, as figure 3.2 shows, the individual words that are instances of the lexeme node LOVE, such as love, loves, loving and loved, create more sub classes/sub nodes of the LOVE node. The same path/value pairs (except the path form and word) of those subclass nodes are inherited from the LOVE node. The nodes have different path form values. Thus their word value is <mor "<form>" >.

Figure 3.2: The morphology of the word love as defined by DATR. Source: Evans and Gazdar (1996)

Consider the node word3 in figure 3.1, which describes the present participle of the verb love. The definition of the present participle form of love is:

word3 :
<>
 == LOVE
<form> == pres part

The value of <word> ( or <mor "<form>" >) at word3 is established by sending up to the hierarchy of nodes, first to LOVE and then to VERB. Here this can resolve <mor "<form>" > by substituting <mor pres part>. Now the value of <word> at word3 has been established by <mor pres part>. Now move up in the hierarchy to VERB and seek the definition for <mor pres part>. Here, <mor pres part> is defined as the sequence of “<root>” ing. This leads us to look for the <root> of word3, which is LOVE. As a result it gives love ing.

The derivation word3 can be shown as follows:
Love is a regular verb. However, not all verbs have the same properties. For example, the past participle of some verbs, such as sew, saw and mow, is the root word with an n attached instead of ed as in regular verbs (see figure 3.3). DATR’s definition by default allows for representation of irregular and subregular lexemes. Irregular behaviour/properties, such as the past participle of verbs ending in n, is represented by a new sub node EN-VERB, which defaults to VERB. Then it overrides the past participle morphology as shown in figure 3.2. This means the path/value pairs of EN-VERBS inherit from VERB, but mor past part is overridden by value “<root>” n.

In summary, Evans and Gazdar explain that morphology, phonology and syntax can be modelled by modelling individual lexical entries using DATR’s language.
description. Furthermore, they show that DATR’s non-monotonicity and inheritance machinery allow it to capture regular patterns of lexemes (love, loving) as well as irregular patterns of lexemes (sew, sewn).

3.3 ELF: The Extended Lexicon Framework

![Diagram of ELF](image)

Figure 3.4: In ELF, words have access to the information of their neighbouring words. Source: [Evans (2013)]

[Evans (2013)] introduced the extended lexicon framework (ELF), a recent direction of development in DATR. ELF uses DATR to represent words not as isolated individuals, but as instances occurring in sentences. Even though information is still represented on a word-by-word basis, the information about a word depends upon information about its neighbours in a sentence.

For example, the representation of the word a might encode the notion that if the next word starts with a vowel, then its form is an instead of a. Let’s consider the following example sentence:

‘A cat saw an elephant.’
The determiners, *a* and *an*, are inherited from one node, having rules for the determiner *a*. Then the vowel *e* in *elephant* changes the determiner *a* to *an*, whereas the letter *c* in *cat* is not a vowel, so it does not change the determiner *a*. That is, in figure 3.4, *word1* and *word4* inherit from the same node for *a*, but have different forms because of the following words in the sentence (*word2* and *word5*). Evans (2013) shows the above example by considering a simple lexical fragment that models the *a/an* behaviour of English, shown in figure 3.5. Evans added two components to this analysis. He defined a feature `<vstart>` at the root CATEGORY node, the value of which is true if the current form starts with a vowel, else it is false. This definition passes by default to all nodes, so every word instance node has the definition. In addition, Evans defined another feature `<form>` in the lexeme node for *A*. But here it is just a definition for overriding *A*. He used the feature `<next vstart>` for a simple table lookup to define a value for the form. This is not defined in the node *A*, because *A* does not define `<next>`, but any words inheriting from *A* assign its `<vstart>` value by following its next inheritance path.
3.4 The Modelling Approach

In the extended lexicon model, a lexicon represents a large set of instances of words. Instances know about their context, especially the words around them. DATR’s default inheritance mechanism allows the lexicon to include special cases, exceptions and example-based analysis where required. These techniques allow the lexicon to encode information about sentence-based natural language processing tasks, such as POS tagging, as lexical knowledge (Evans, 2013).

Parts of speech (POS) is a category of words that have similar properties within the grammatical structure of sentences. POS play an important when a group of words are joined together and make a meaningful sentence. A POS tagger is a piece of software that tags POS in words in a corpus based on its context and definition. Brill (1992) tagger was one of the first POS taggers and has been commonly used in NLP. There have also been some other POS taggers such as the Stanford Log-linear POS tagger (Toutanova and Manning, 2000), the Tree tagger (Schmid, 1995), and the Microsoft POS tagger (Kim et al., 2015).

Typical POS taggers are modelled using either rule-based algorithms or statistical algorithms. Brill (1992) used rules or transformations to create a POS tagger. Brill used an annotated training corpus for the system to learn the grammar rules by deriving lexical/morphological and context information, but without any expert knowledge or human interaction. Brill’s rule-based POS tagger extracted information from the training dataset using statistical techniques. Then, Brill used a program for learning the grammatical rules. He did not use any language-specific knowledge. However, Evans (2013) showed that POS tasks can be constructed without rule-based or statistical techniques, but as lexical description tasks. Evans used the feature <pos> for each word instance, with a definition provided at the root node. It defines the value of the feature <pos>, based on the <pos> of the previous two words. In this way, Evans showed that grammatical components of language processing can be reduced and substituted by the structure of ‘putting words next to each other’ in a sentence.

Similarly, statistical algorithms have been used for POS taggers in previous studies (Garside, 1987). Statistical techniques can be very useful for modelling messy systems in language processing tasks. However, it is not always an effective approach to exploit context information or lexical knowledge. Evans (2013) showed that the extended lexicon can change language processing tasks into a lexical system, rather than an external algorithmic component, and that some of the default mechanisms in DATR/ELF have a similar descriptive effect to statistical operations such as backing off.
The main advantage of lexical description languages such as DATR is their method of organising information, and the fact that they allow scholars to arrange regularities as well as irregularities in order. Moreover, DATR/ELF is beyond simple ‘lexical’ representation: it extends to sentence-level linguistic representation. Accordingly, this lets me replicate the algorithms and techniques used in the previous lexicon-based approaches for sentiment analysis in a single lexical modelling framework, and then allows me to extend the analysis to include inheritance, larger elements of sentences (phrases, syntax, etc.) and example-based learning additions.

3.5 The ELF Implementation

ELF is implemented as a DATR library by using special built-in functions to allow it to ‘compile’ new DATR nodes dynamically for any given sentence, and then discard them at the end of processing. This is because an ELF lexicon is effectively infinite (as it contains a node for every word instance of every possible sentence). What I define is a core lexicon, which is similar to a traditional DATR lexicon, and then the system adds ‘instance’ nodes for a particular sentence dynamically. These nodes are linked to corresponding abstract lexical nodes in the core, and also linked to each other using <prev> and <next> links to form a sentence. It is these links that the abstract node definitions exploit to allow a word instance to access sentential context. Information about a sentence can then be obtained by querying these instance nodes.

Furthermore, ELF provides a mechanism for separating lexicons into separate components (‘models’ or ‘layers’) that divides the definition of lexical items into different groups. ELF models can be inherited from each other, which allow me to build an incremental model with more functionality through a sequence of ELF models. For example, in the ChartEx project\(^\text{2}\) ELF is used to implement the parsing engine Celeborn, which has models for token-level, lexical, phrasal and sentential analysis. In the present project, I used ELF models in a different way, to implement different sentiment models (rules and algorithms to calculate sentiment scores), each one inheriting from the previous one and adding complexity.

I started with a simple sentiment analysis engine and incrementally developed a sentiment analysis system called Galadriel (see chapter 4) that runs in Windows and Linux. The output is in the form of BRAT \(\text{Stenetorp et al., 2012}\) annotation data. ELF also does the pre-processing such as tokenising, POS tagging, stemming, parsing, etc., within the framework. This allows my sentiment analysis engine to

\(^2\text{https://www.chartex.org}\)
access more formats than just a word format such as word stem, POS, etc. I also used several ELF built-in functions, such as ‘Eval’, which does more complicated sums using simple arithmetic expressions $+, -, *, /, **$ and double brackets, IF-ELSE statements and OR statements.

However, there are some limitations of the DATR/ELF implementation too. Unlike other object-oriented languages, DATR does not handle any cyclic references. Even though it is theoretically possible to write cycle references in DATR, there would be limited uses for this. So, currently, cyclic references in lexical specification cannot be resolved within DATR/ELF, and may cause infinite loops. Moreover, the DATR implementation is not designed for dealing with reverse inferences (mapping from values back to node/path combinations). However, reverse inferences are not required for sentiment analysis. Another drawback of the DATR language is that it does not use a typed feature structures. Thus it is not as efficient and compact as the languages that do use a typed feature structure. DATR/ELF models cannot deploy machine learning techniques by automatically acquiring rules from annotated corpora. However, there is the potential for building robust extended lexicon models in the future. In this thesis, I attempted to exploit the corpus-learning methodology to populate lexical entries from examples and add them to the sentiment lexicon manually but not automatically.

### 3.6 Summary of the Chapter

This chapter has provided a review of DATR/ELF and compared it with the other representation languages and the other grammar frameworks. I further provided a self-contained summary of the algorithms used within DATR/ELF. The chapter also compared the inheritance-based lexical knowledge modelling approach with other similar approaches by using an example modelling of a POS tagger. I then provided a comparative critique of the suitability of DATR/ELF implementation for sentiment analysis. Finally, I produced a brief introduction of my sentiment analysis research tool *Galadriel*, which is based on a DATR/ELF framework.
Chapter 4

Overview of Galadriel

The lexicon-based method is one of the main approaches in sentiment analysis. As it is an unsupervised approach, it does not require large datasets. Another of its main strengths is that it can take advantage of context analysis to determine the sentiment of more complex statements. However, a lack of robustness and necessity of expert manual construction of the lexicon are challenges for the lexicon-based approach. This project aimed to solve these issues by using non-monotonic (default) inheritance networks. My intuition is that sentiment analysis systems can be made much more practical by explicitly considering lexical knowledge representation. In this chapter, I start with the problem statement, which gives a clear description of the problem being addressed. Subsequently, section 4.2 presents the motivation for the inheritance modelling. This section focuses on the basic theoretical framework of an inheritance-based lexicon for sentiment analysis. I then introduce my research tool Galadriel, which uses the language for lexical knowledge representation, DATR, and the extended lexical framework (ELF), which I discussed in chapter 3. This chapter outlines the key building blocks of Galadriel.

4.1 Problem Statement

In chapter 2, I discussed various approaches that have been used for sentiment analysis in recent years. One of the major advantages of the lexicon-based approach is that it can successfully handle contextual valence shifters such as negation (e.g. not good) and intensification (e.g. very good). Recent lexicon-based systems are reporting better performance figures than machine-learning systems. However, this approach also has significant shortcomings. It relies on specific opinion lexicons: when a piece of text contains an unknown word (i.e. a word which is not in the
lexicon), the lexicon-based approach may produce a wrong result. Usually, it classifies the word as neutral as the word is defined as neither positive or negative in the opinion lexicon. Lexicon-based approaches are also based on specific rules, but there are many cases where the rules do not work properly, and other rules could be applied. The lexicon-based approach is not very robust or systematic because it cannot cope with errors during execution. That is to say, lexicon-based models do not work well with invalid and unexpected inputs. To overcome these problems, I aimed to develop novel techniques that accommodate different rules in order to handle various types of circumstance. I also attempted to employ corpus patterns from the example-based learning method, by using non-monotonic reasoning to override or modify rule-based behaviour. My intuition is that sentiment analysis systems can be made much more effective by explicitly considering more advanced lexical knowledge techniques using non-monotonic (default) inheritance networks. For this reason, I aimed to investigate how inheritance-based modelling techniques can perform sentiment analysis and to demonstrate the added value of this novel method.

4.2 Motivation for Inheritance-Based Modelling

Words have different kinds of impact on sentiment. Some of them have no impact, others have a lot of impact, and they can be grouped according to their ‘sentiment behaviour’, and then subgroups can be identified with more specific similarities.

A sentiment lexicon is thus a set of lexical entries in which each lexical entry has its own sentiment behaviour. More than one lexical entry or group of lexical entries might share the same sentiment behaviour. Other lexical entries share only some sentiment behaviour. For example, let’s consider a set of lexical entries $L = \{\text{happy, good, nice, excellent, bad, sad, ugly, man, cat, tree, walk, talk, brief, average, typical, common}\}$. These lexical entries can be divided into two groups depending on their sentiment behaviour. That is, $\text{happy, good, nice, excellent, bad, sad and ugly}$ express some sentiment behaviour, and this can be made subset $S$ of $L$, whereas $\text{man, cat, tree, walk and talk}$ do not show any sentiment and can be identified as subset $N$ of $L$. The elements of $S$ share certain properties. Hence, lexical entries of subset $S$ have the same sentiment behaviour. Thus the distinction between subsets $S$ and $N$ are having two different kinds of sentiment behaviour, which are words with sentiment and words without sentiment. Figures 4.1a and 4.1b show the main set and its subsets. To illustrate, the lexical entries of the superset $L$ have a feature, ‘if the lexical entry has sentiment or not?’, which can be transformed into a hierarchical structure (see figure 4.1). Abstract node $L$ has a feature, ‘has-sentiment’. Subset $S$
(a) Example lexical entries in a given set

(b) Subsets of L are named S and N

(c) Subsets S and L can be explained in a hierarchical structure

Figure 4.1: Set L is divided into subsets S and N and explained in an inheritance structure
has the value ‘True’, and subset \( N \) has the value ‘False’.

The properties of the entire set \( L \) can be described in a root node \( L \) as shown in figure 4.1c. Then the properties of subsets \( S \) and \( N \) inherit from node \( L \). Finally, lexical entries inherit from each abstract node \( S \) and \( N \).

![Figure 4.2: Two further subsets of \( S \) are created](image)

Subset \( S \) contains both positive and negative lexical entries. So, it can be divided into two further subsets, depending on its lexical entries’ sentiment properties, and explained in an inheritance structure, as shown in figure 4.2. Thus nodes (subsets) \( POS \) and \( NEG \) have the same properties as their superset \( S \), which are expressing sentiment and having a polarity value (non-zero value): node \( POS \) has a positive polarity value and node \( NEG \) has a negative polarity value. This inheritance structure helps to capture generalisations of the lexical entries.

This inheritance structure can get more complicated, as the \( POS \) and \( NEG \) nodes can be furthermore divided into subsets depending on the strength of their sentiments (positivity and negativity). Besides, some neutral lexical entries modify their nearby sentiment lexical entries, such as intensifiers and negators. Therefore, the neutral class can also be divided into further subclasses.
4.3 The Research Tool: The *Galadriel* system

This project aimed to exploit lexicon representation that can encode very complex information such as phonology, morphology, syntax and semantics for sentiment analysis. I used the DATR representation language, which supports the notion of inheriting lexical information from abstract classes, but also the possibility of overriding inheritance. In order to make use of the DATR language, I started out with an initial *Galadriel* system (*Galadriel* 0.1), which is a very simple sentiment framework using DATR/ELF (see figure 4.3).

![Figure 4.3: The skeleton of *Galadriel* 0.1 framework](image)

4.3.1 *Galadriel* Models

The *Galadriel* sentiment model is lexically based, so all the generalisations (that is, the sentiment model) are in a single abstract lexeme node which all words inherit from, with each word operating as an independent lexical agent to calculate sentiment scores incrementally. It is assumed, in the basic *Galadriel* architecture, that each word (independent lexical agent) has two feature:value pairs, and the two features are score and total. All the lexical agents for actual words inherit from an abstract lexical agent node called lexical-agent1. This node specifies the default values for the (feature) score of 0 (neutral) and overrides with its own (base) value, which can be imported from *Galadriel*’s sentiment lexicon. (I discuss the modelling of the sentiment lexicon in the next section.) The node also has rules for calculating the feature total, by adding the score to ‘prev total’, the total from the previous word. Consider the following example:

‘It is a very good movie’

Figure 4.4 shows that, in the basic *Galadriel* architecture, all the word nodes inherit both these specifications, except the word *good*, which specifies its own score of +3,
Figure 4.4: Simple sentiment model: adds up raw sentiment score of all words and produces total sentiment score

Figure 4.5: Sentiment model with intensifiers: *very* changes sentiment score of following word
overriding the (default) inheritance from lexical-agent\textsubscript{1}. The resulting values for score and total are shown in the figure, and the sentiment score for the whole sentence can be read off from the value of the total feature for the last word.

In figure 4.5 I extend this model with another agent, lexical-agent\textsubscript{2}, which describes a rule for intensifiers. This rule says that if the previous word is \textit{very}, then this word’s sentiment score has to be multiplied by a factor of 2. In this example, lexical-agent\textsubscript{2} is only used for sentiment-bearing words, such as \textit{good} – neutral words simply inherit from lexical-agent\textsubscript{1} as before. Therefore, the sentiment score of \textit{good} changes, and all other words’ scores remain as before.

Similarly, consider another example:

\begin{quote}
‘The movie is not good.’
\end{quote}

The sentiment word, \textit{good}, in the above example has to be negated by the previous word, \textit{not}. As a result, the positive word, \textit{good}, becomes negative. So I extend the model with a rule for negation, that if the previous word is \textit{not}, the word is multiplied by -1.

However, the word \textit{not} is not necessarily always present immediately before the sentiment word in the sentences. Consider the following sentence;

\begin{quote}
‘This movie is not a good movie.’
\end{quote}

In order to overcome this issue, I introduce another feature, \texttt{neg-context}, that can take the values either \texttt{yes} or \texttt{no}. Then I extend lexical-agent\textsubscript{2} with lexical-agent\textsubscript{3}, with the updated negation rule that assigns a value for the \texttt{neg-context} of the word: \textit{not} is \texttt{yes}, and for other words, it is assigned its previous \texttt{neg-context} value, as shown in 4.6.
4.4 Modelling of Sentiment Lexicon

This section shows the modelling of the sentiment lexicon or sentiment dictionary in Galadriel ‘base model’. The base model here is the simple model which just gives scores to lexical items and does not try and calculate anything and the other sentiment model sits on top of it, using base scores as a default value for its scores. As I described before, each lexical item operates as a lexical agent (or automaton). Each lexical agent has a set of feature:value pairs, which define its sentiment behaviour. I group these into categories depending on their sentiment behaviour, and they are structured in an inheritance network, in which the nodes describe the sentiment behaviour of the categories in the Galadriel base model. The set of features and their possible values that I use are described in the next section.

4.4.1 A Feature-Based Model of Sentiment Behaviour

- **type**: This feature indicates the sentiment class of a group of lexical items. The common possible values are positive, negative and neutral.
• **neg-context**: This feature is used to identify if the lexical item is in a negation context. The value takes either *yes* or *no*.

• **score**: This is the most important feature for the lexical item in any sentiment analysis task, as it decides the final sentiment of the lexical item. The value of the feature is a positive (+) or negative (-) real number. The sign of the value indicates whether the word is positive or negative and the number shows the magnitude, i.e. how positive/negative the word is. The value of any neutral words takes 0.

• **total**: The feature **total** of the lexical item indicates its total sentiment score within the sentence. This is the total sentiment score for the sentence up to and including the current lexical item. The following equation is used for calculation:

\[ \text{total} = \text{score} + \text{prev total} \]

### 4.4.2 Inheriting Sentiment Behaviour

As I explained before, every lexical agent (item) has the same set of features with different values, depending on its sentiment behaviour. Figure 4.7 shows lexical items structured into a basic inheritance network, with abstract nodes which share feature values. In this section, I discuss the nodes of the lexicon-based inheritance structure and their sentiment behaviour.

#### 4.4.2.1 SENTIMENT Node

All lexical items can be described in an inheritance hierarchy based on their sentimental features, described by the nodes in the hierarchy. I create a root node called SENTIMENT that explains the general (default) sentiment behaviour of all lexical items. I use the above feature model to describe the sentiment behaviour. The SENTIMENT node has three children (polarity) nodes, representing sentimental polarity categories POSITIVE, NEGATIVE and NEUTRAL. Each polarity node passes down their sentiment behaviour to appropriate lexical instances, which are at the lower level. I start at the SENTIMENT node, which assigns the following default values for the above features:
Figure 4.7: The lexical items are structured into a tree using abstract nodes

\[
\begin{align*}
type &= \text{neutral} \\
neg-context &= \text{no} \\
block-context &= \text{no} \\
\text{score} &= 0 \\
\text{total} &= \text{score} + \text{prev total}
\end{align*}
\]

Then the abstract nodes POSITIVE and NEGATIVE inherit from the abstract SENTIMENT node, which is the root node in the higher-level structure. Thus, the feature values of SENTIMENT inherit to the POSITIVE and NEGATIVE (or polar) nodes, which have their own sentiment behaviour too. In particular, the values of type and score are overridden by new values. For example, in the POSITIVE node, the following values are overridden and the other values remain the same:

\[
\begin{align*}
type &= \text{positive} \\
\text{score} &= +1
\end{align*}
\]

Furthermore, as figure 4.7 shows, positive word lexical agents (happy, good, nice,
etc.) inherit from the POSITIVE node, and all the feature values of the POSITIVE node are passed down to the positive word lexical agents, except the value of score, as each word has its own sentiment score.

**NEUTRAL node:** As shown in figure 4.7, as the default feature values for type and score of SENTIMENT node are neutral and 0, any non-polar (non-positive or negative) words are classified as neutral. I thus created a node NEUTRAL, which is another subclass (node) of SENTIMENT, and its feature values inherit from the SENTIMENT node. Sentiment neutral words are words that do not show any sentiment at the word level, which means their sentiment score is 0.

However, this is not always true for sentence- and document-level analysis tasks. Neutral words can be exploited in various interesting ways in sentiment analysis tasks. Consider the words very and not in the previous examples. They are neutral words, but they change the sentiment score of their neighbouring words in different ways. Similar to very, other intensifiers such as really, slightly, etc. also change the sentiment score of their neighbouring word by changing its intensity/magnitude. They were grouped together and modelled under the node called INTENSIFIER, which is inherited from NEUTRAL. On the other hand, some negators change the sentiment score of their neighbouring lexical items by changing their polarity. Another important subclass of NEUTRAL is punctuation marks (BOUNDARY), which

---

Figure 4.8: The NEUTRAL node and its subclasses in the inherited hierarchy
define the boundary of sentences or clauses. However, some other ordinary neutral
words, such as *man, house, etc.* are not involved in any sentiment analysis tasks. I
have identified different subclasses of neutral words depending on their behaviour,
and structured them in the inheritance hierarchy under the abstract node NEUTR-
RAL, as shown in figure 4.8. The feature values of each subclass inherit from
NEUTRAL, but some features are overridden by their own values. It is also possi-
bile to introduce more feature values depending on the sentiment behaviour of lexical
items and models in an appropriate node in the *Galadriel* inheritance-based lexicon.

### 4.5 Basic *Galadriel* Code

```
galadriel.sentiment.ROOT: <> == galadriel.LEXROOT

galadriel.sentiment.POSITIVE: <> == galadriel.sentiment.ROOT:<>
  <base type> == positive
  <base score> == 1

galadriel.sentiment.NEGATIVE: <> == galadriel.sentiment.ROOT:<>
  <base type> == negative
  <base score> == 1

galadriel.word.N-happy: <> == galadriel.sentiment.POSITIVE.
galadriel.word.N-good: <> == galadriel.sentiment.POSITIVE.
galadriel.word.N-nice: <> == galadriel.sentiment.POSITIVE.
galadriel.word.N-sad: <> == galadriel.sentiment.NEGATIVE.
galadriel.word.N-bad: <> == galadriel.sentiment.NEGATIVE.
galadriel.word.N-horrid: <> == galadriel.sentiment.NEGATIVE.
```

Figure 4.9 shows the basic *Galadriel* with inheritance of the sentiment lexical infor-

---

*Figure 4.9* shows the basic *Galadriel* with inheritance of the sentiment lexical infor-
mation represented in DATR. The figure shows the sentiment information only in terms of polarity. The sentiment information of intensity (or magnitude) could also be included.

4.6 Galadriel’s Output

Figure 4.10 shows Galadriel’s input and output for a Yahoo customer review. A plain text file was used as an input document for analysis in the Galadriel system. The output is in the BRAT format, and gives a breakdown of the document with the appropriate feature values. In figure 4.10, the Galadriel output text shows the total value of the last word of the input text, which is the total score for the document/sentence. The example output document shows the neg-context, score and total values. However, this can be changed by changing the Galadriel output setting.

![Input text](image1)

![Galadriel output](image2)

Figure 4.10: The Galadriel input and output text for a Yahoo restaurant review document
Figure 4.10 shows the first couple of words (top part) and the last few words (bottom part) of the document. In the top of the output, it shows which model (sent3 or a model with lexical-agent3) was used to get this output. Then each word has a ‘Txx’ identifier line for each word and its type followed by ‘Axx’ identifier lines for each output feature and its value. All feature values and their calculations are inherited from the lexical-agent3 which is inherited from lexical-agent2. The total score for the document is shown by the total score value of the last word. (It is -7.75 in the above example.)

4.7 Discussion

In general, basic lexicon-based approaches consider a text as a group of words that are arranged in an order, and the words are assigned a sentiment value using a sentiment lexicon or dictionary. The approach calculates the overall sentiment score of the text by aggregating the sentiment value of each word in the text by taking account of contextual information or valence shifters, such as negators and intensifiers. In the Galadriel system, I use a base model to model the basic static sentiment lexicon. DATR’s inheritance mechanism then allows me to model more advanced sentiment lexicons using a default inheritance structure. This provides the benefit of reusability for the sentiment behaviour of lexical items. I collected the same behaviour of sentiments or similar sentiment values of lexical items modelled under an abstract node of the inheritance structure. I then assign a set of Galadriel feature values to the abstract node that are passed down to the lexical items, which also can be overridden by their own feature value. Moreover, the feature values of each lexical item of the Galadriel lexicon can also be overridden based on contextual information. This is one of the efficient features of the Galadriel system. Moreover, typical lexical-based approaches apply different rules and algorithms to handle valence shifters for sentiment analysis. In Galadriel, it is very easy to model different rules and algorithms in different models, and to structure them in a single inheritance-based framework.

4.8 Summary of the Chapter

This chapter started with the problem statement, followed by my motivation for using inheritance-based modelling for sentiment analysis. Then the chapter provided an introduction to Galadriel 0.1, the initial sentiment framework which uses the DATR/ELF representation language. The key features of Galadriel were then
introduced. I first discussed how the Galadriel system considers each word as a lexical-agent with two feature: value pairs (score and total) which inherits from a lexical agent node that specifies the rules to assign and calculate values for each feature. I showed how the lexical agent node can be extended with additional rules for the valence shifters (negation and intensification) and how the feature values are calculated. I also provided basic Galadriel code that represents and inherit the lexical information using the DATR language. Finally, I provided an example of Galadriel’s output and explained the final score and how it was calculated. The next chapter discusses how existing lexical approaches can be implemented in Galadriel. Then chapter 6 develops the final version Galadriel 1.0 using various features, including the existing approaches.
Chapter 5

Representative Lexicon-Based Approaches

As stated in chapter 1, my first research objective was to model sentiment knowledge using DATR’s inheritance mechanism, by modelling existing lexicon-based approaches to sentiment analysis. Therefore, I aimed to study two existing lexicon-based approaches used for sentiment analysis and replicate their work in my system. This chapter assesses the two different lexicon-based methods which were chosen, and I provide a comparison between them and an evaluation of each method. The first method is Taboada et al. (2011) study, ‘Lexicon-Based Methods for Sentiment Analysis’. The second study is ‘A Holistic Lexicon-Based Approach to Opinion Mining’, by Ding et al. (2008). I chose these two methods because most of the lexicon-based methods for sentiment analysis done in recent years are based on these two approaches. Moreover, these methods offer us two unique perspectives into the phenomenon of the lexicon-based approach to sentiment analysis. However, their research aims, strategies and methods are significantly different from each other: Taboada et al.’s lexicon method is a document-level sentiment analysis system, whereas Liu et al.’s holistic approach is a feature-level (or aspect-level) sentiment analysis system.

Taboada et al. (2011) work involved developing a sentiment analysis system to calculate the total sentiment of a text using dictionaries of words annotated with their semantic orientation, which is a combination of the word’s polarity and its intensity (strength). A semantic orientation (SO) score for a word/text decides whether it is positive or negative depending on its sign, and it decides its sentiment intensity depending on its value. (SO value of +5 is the strongest positive, whereas −5 is the strongest negative.) They used a simple lexicon method, that is, calculating the semantic orientation of a text by aggregating the semantic orientation of each opin-
tion word present in the text. They also added some features for handling valence shifters (intensifiers, negation and irrealis markers) which may potentially change the semantic orientation of the words present in the text. To calculate the semantic orientation of words, they developed a system called SO-CAL. This is an extension of their previous version, which contained only an adjective dictionary and basic calculation, such as aggregation and averaging semantic orientation. Ding et al. (2008) also wanted to extract opinion/sentiment from a text. Their research involved feature-/aspect-level sentiment extraction, which was slightly different from Taboada et al. (2011). They wanted to analyse sentiment in a text/review/blog regarding particular product features. As a typical lexicon-based method, Ding et al. (2008) also used opinion-bearing words (or simply ‘opinion words’) to analyse the sentiment of the text. They counted the number of positive and negative opinion words that were present near the product feature words to decide whether the text had a positive or negative sentiment towards the product feature. Unlike Taboada et al., Liu et al. aimed to find out only the polarity of a text using the polarity of words present in the text. They did not consider the intensity of the word or the text. Moreover, they proposed a technique to deal with context-dependent opinion words, which are non-opinion words that can express opinion depending on their context. They built a system called Opinion Observer (OO), and their approach exploits external information and evidence by using linguistic conventions to handle those context-dependent opinion words.

In this chapter, I briefly describe SO-CAL and Opinion Observer, and the features used in their heuristics. Sections 5.1.1 and 5.3.1 describe how I modelled each system in Galadriel. I also present an evaluation which shows how Galadriel’s performance compares with each system and I introduce a new calibration method to tune the system to maximise performance.

5.1 The SO-CAL System

This section begins with an overview of Taboada et al. (2011)’s SO-CAL system. Then I show how SO-CAL features were modelled in Galadriel. To propose a method for sentiment analysis, Taboada et al. (2011) aimed to analyse the semantic orientation of individual words and contextual valence shifters in depth. However, they did not focus on linguistic analysis. First, they extracted sentiment-bearing words from a document, including adjectives, adverbs, nouns and verbs. Then they used the semantic orientation score from their semantic orientation dictionaries to calculate a score for the whole document, taking into account valence shifters such as intensifiers and negators. Semantic orientation dictionaries are specialised dictionaries
which include words with their semantic orientation. Lexicon-based methods use these dictionaries, and they can be created in different ways. Taboada et al. created their dictionaries manually, as they believed that the method of creating dictionaries affects the overall accuracy of the final results. Then they applied the SO-CAL calculator to calculate the overall semantic orientation (SO) value. Similar to most of the previous lexicon-based methods \cite{Bruce and Wiebe, 1999; Kim and Hovy, 2004a; Hu and Liu, 2004}, Taboada et al. \cite{2004} adopted two assumptions in their approach. The semantic orientation of a word is independent of its context, and this semantic orientation can be expressed in a numerical value. The following sections will explain how they created their semantic orientation dictionaries for the words (including valence shifters) and how their SO-CAL system works.

5.1.0.2 Adjectives, Nouns, Verbs and Adverbs

In previous versions of SO-CAL, Taboada and Grieve \cite{2004} and Taboada et al. \cite{2006} only focused on adjective words. An adjective dictionary was used before, which was hand-tagged on a scale ranging between $-5$ (strongest negative) to $+5$ (strongest positive). However, this SO-CAL system aimed to take the other parts of speech into account\footnote{1} in order to produce a more sophisticated system.

For this approach, Taboada et al.’s previous adjective dictionary was extended with more SO carrying words. So, additionally, opinion adjectives were taken from different sources and ranked by hand for the new SO-CAL version. The sources were: a 400-text collection of eight different domains \cite{2004}, positive and negative words from the General Inquirer dictionary \cite{Stone et al., 1966}, and 100 text movie reviews \cite{Pang et al., 2002}. The dictionaries for nouns, verbs and adverbs were also created using the above sources and hand ranked on the same scales, i.e. from $-5$ to $+5$, as the adjective dictionary. Any words not assigned an SO value were hand ranked by a native English speaker. The SO values for adverb words were generated by matching the SO value of corresponding adjective words. When SO-CAL wants to calculate the SO value of an adverb in a text, and it is not found in its dictionary, SO-CAL considers its stem and matches the corresponding adjective word to obtain the SO value. All nouns and verbs were added in lemmatized form, and the word form was not taken into account. Finally, an enhanced dictionary containing 2252 adjectives, 745 adverbs, 1142 nouns and 903 verbs, with SO values between $-5$ (extremely negative) and $+5$ (extremely positive) was produced and validated.

\footnote{1}{The Brill tagger \cite{Brill, 1992} was used to determine parts of speech}
5.1.0.3 Intensification

Taboada et al.’s dictionary of intensifiers contains 177 entries, including some multiword expressions, with positive and negative percentage values. The intensifiers with a positive percentage are categorised as amplifiers, which increase the semantic intensity of a neighbouring lexical item, whereas intensifiers with negative percentage values are downtoners, and decrease it. Those percentage values explain what percentage of the word’s SO value has to be modified. To get the final SO value of a word with any intensifiers, SO-CAL modifies the SO value of the word by the associated percentage value of it. For instance, consider the following intensifiers with their percentage values:

‘somewhat’ = −30%
‘most’ = +100%

If sleazy has the SO value −3 and excellent has the SO value +5, then to calculate the SO value for somewhat sleazy, the SO value of sleazy needs to be modified by -30% of -3, which is:

\[3 \times (-30/100) = +0.9\]

The SO value of somewhat sleazy is thus:

\[-3 + 0.9 = -2.1\]

To calculate the SO value for most excellent, the SO value of excellent needs to be modified by +100% of +5, which is:

\[+5 \times (+100/100) = +5\]

The SO value of most excellent is thus:

\[+5 + 5 = 10\]

Furthermore, the same process was applied to other parts of speech, such as adverbs and verbs with SO values. Also, some SO-valued nouns (e.g. failure) could be modified by adjectives (e.g. total), which are called adjectival intensifiers. SO-CAL has a separate dictionary for adjectival intensifiers. Moreover, another three categories of intensification were added to the system. These are the usages of all capital letters, exclamation marks and the discourse connective but, which are used to explain noticeable information.
5.1.0.4 Negation

Saurí (2008) produced a switch negation method to deal with negation words. Switch negation simply reverses the polarity (the sign of the SO value) of the lexical item next to a negator found in a text. The same approach was used in SO-CAL but with added techniques. The SO-CAL system focused on negation words, including not, nothing, never, etc., and some verbs and nouns, such as without, lack, etc. Furthermore, a polarity shift method was implemented instead of just switching the polarity (switch negation) of the word, as the switch-negation method fails in most of the situations (Liu and Seneff, 2009), because simply reversing the polarity of a word for its negation does not give the correct strength of the polarity.

Examples:

\[
\begin{align*}
\text{terrible} &= -5 & \rightarrow \text{not terrible} &= +5 \\
\text{excellent} &= +5 & \rightarrow \text{not excellent} &= -5
\end{align*}
\]

In these examples, if terrible is negated by the switch negation method, then the SO value of not terrible will be +5, which is not true, as not terrible is not a strong positive phrase. Similarly, not excellent is not a strongly negative phrase. But the switch negation method makes it a strong negative.

The polarity shift method moves the SO value of the word/phrase which needs to be negated towards the opposite polarity by a fixed amount 4. Examples:

\[
\begin{align*}
\text{terrible} &= -5 & \rightarrow \text{not terrible} &= -5 + 4 = -1 \\
\text{excellent} &= +5 & \rightarrow \text{not excellent} &= +5 - 4 = +1
\end{align*}
\]

Moreover, it has been argued that negation words do not only change the polarity of the word next to them. In some cases, they negate other words present within the same clause. Some techniques have been added to SO-CAL to capture boundary clauses in a sentence. These techniques include a search for the clause boundary marker and a search for skip words for parts of speech to separate the boundaries in the text.

5.1.0.5 Irrealis Blocking

Another list of words was introduced and referred to as ‘irrealis’ markers. This list includes negative polarity items (anything, any...), conditional markers, modal verbs (should, could...), some intentional verbs (expect, doubt...) and questions (?).
The sentences with those words do not necessarily reflect the author’s opinions, so those sentences are not reliable for sentiment analysis. Moreover, those words are usually used in non-factual contexts. The SO-CAL system ignores any sentiment orientation of a word within the clause if an irrealis marker is found.

Examples:

‘This should have been a great movie’ − +3 → 0
‘This movie could be one of the best of the holiday season’ − +5 → 0

However, it is not necessarily the case that all questions act as irrealis markers, and some questions do reflect the sentiment of the text. To identify such cases, SO-CAL checks if any determiners are present before the opinion words in the sentences. In such cases, SO-CAL ignores the irrealis marker, the question mark at the end of the sentence.

Example:

‘he can get away with marketing this amateurish crap and still stay on the bestseller list?’

In the above sentence, this blocks the irrealis marker ?.

5.1.0.6 Text-Level Features

Taboada et al. (2011) believe that human language mostly favours positive language. As a result, the lexicon-based classifiers mostly show a positive bias. To overcome this problem, they added a cognitive weight to all negative expressions. Thus, in SO-CAL, the final SO value of any negative expression is increased by 50%.

Moreover, SO-CAL wants to avoid the repetition of a word adding more sentiment weight to the sentences. Accordingly, if a document/review contains a word more than once (say n times), then the SO value of n'th appearance of the word has been assigned $1/n$ of the full SO value of the word in SO-CAL.

Example:

‘Overall, the film was excellent, the acting was excellent, the plot was excellent and the direction was just plain excellent.’

The word excellent appears four times in the above sentence. This does not mean that the sentence expresses a very strong positive sentiment. For instance, the SO
SO-CAL Features | Calculation of SO value
--- | ---
Adjectives, adverbs, nouns, verbs (SO value: a, b, c, d, ... ) | \( a + b + c + d \ldots \)
Intensification (factor: \( P \% \)) | \( a + (a \times P/100) \)
Negation (a fixed amount 4) | \( a - 4 \) if \( a > 0 \)
Irrealis blocking | \( a = 0 \)
Negative expressions | \( \sum (a + b + c + d\ldots) + \sum (a + b + c + d\ldots)/2 \) if \( (a + b + c + d\ldots) < 0 \)
Repetition | \( a + \ldots + a/2\ldots + a/3\ldots a/n \)
\( n \) is number of appearance of the word w
\( a \) is the SO value of word w

Table 5.1: Summary table of SO-CAL features and their calculation methods.

value of each instance of excellent in the above example will take 5, 2.50, 1.67, 1.25, respectively, and the sentence gets the total SO value of 10.42, instead of 25.

5.1.0.7 Other Features of SO-CAL

SO-CAL uses another two features, which are derived from external sources. These features are not appropriate for sentiment analysis purposes, and thus I did not use the following features in Galadriel.

**Weighting:** An XML weighting option was introduced as another feature of SO-CAL, and allows the system to give extra weight to a portion of a text or sentences. Firstly, topic sentences in a text are identified by pre-processing. Then, the topic sentences are tagged using the XML weighting option. Thus, any words between these tags are multiplied by a certain given weight.

**Multiple cut-offs:** Outputs of SO-CAL are numerical values which indicate both the polarity and intensity (strength) of words present in a text. The numerical values are indefinite, which is not a feasible way to produce a clear output. For instance, when customer reviews are assigned a star rating, it is hard to categorise the indefinite numerical values into four or five stars. To overcome this problem, the authors added another feature to SO-CAL, which is multiple cut-offs. This allows SO-CAL to take a list of \( n \) cut-off values, and then classify texts into to \( n + 1 \) classes based on the values. I achieved a similar effect as a post process which calibrates Galadriel’s performance as a classifier using training data (see section 5.6).

Table 5.1 shows a summary of SO-CAL features and their calculation methods.
5.1.0.8 Evaluation of SO-CAL

Evaluation of SO-CAL was carried out in three different stages. First, the authors evaluated SO-CAL features by comparing the performance of the full SO-CAL system with various dictionary alternatives: a simple dictionary, an adjective-only dictionary and a one-word dictionary, in different domains such as books, cars, computers, cookware, hotel movies, music, phones and cameras. The simple dictionary is the simple version of the current dictionary, and includes only values between –2 and +2, intensification factors +1 and –1, and switch negation. The adjective-only dictionary includes only main adjectives. The one-word dictionary excludes multi-word expressions. All the outputs (positive/negative) of the dictionaries were compared to the ‘recommended’ or ‘not recommended’ field of the reviews. This showed the performance of full SO-CAL better than other alternatives, and 78.74% of the corpus returned correct outputs.

Secondly, they tested SO-CAL in other domains. They chose four different datasets, the Multi-Perspective Question Answering (MPQA) Corpus from Wiebe et al. (2005), a collection of Myspace comments from Prabowo and Thelwall (2009), a set of news and blog posts from Andreevskaia and Bergler (2008), and a set of headlines from Strapparava and Mihalcea (2007), and SO-CAL showed a minimum of 75-80% accuracy.

Finally, Taboada et al. validated the SO-CAL dictionaries by comparing them with other existing dictionaries (such as Google, SentiWordNet, Maryland, GI, etc.) and determined that the SO-CAL dictionaries are reliable and robust.

5.1.1 Modelling SO-CAL in Galadriel

In order to test out the Galadriel system’s architecture, I first modelled SO-CAL in Galadriel. In this section, I describe the key steps of the modelling process. I ended up creating a total of six models in Galadriel for SO-CAL features. Each model is used to capture one feature of SO-CAL. In Galadriel, I named the models sent1, sent2 and so on. These models (sent1, sent2, etc.) are actual example of lexical agent models, which were described in chapter 4.

5.1.1.1 Model sent1: Aggregating SO scores

I had four different dictionaries (used for SO-CAL) for the parts of speech, adjectives, adverbs, nouns and verbs, with their SO values (between +5 and –5). As discussed above, in order to get the total SO value of document, SO-CAL aggregates the SO
value of each word present in the document. In \textit{Galadriel}, the model sent1 is a simple model where each word is associated with its own SO score and a total score for the document up to that point. This is as shown in figure 5.1.

\subsection*{5.1.1.2 Model sent2: Intensification}

Intensifiers do not contribute to the propositional meaning of a clause, and they generally do not have any sentiment of their own. But they give additional emotional context to a word they modify, which means intensifiers change the semantic intensity of that word. The words whose SO values are being modified by intensifiers are usually their neighbouring lexical item. \cite{Taboada2011} represented the value of an intensifier as a percentage, and these values are listed in the SO-CAL dictionaries. Figure 5.2 shows the modelling of intensifiers, which uses the same approach as explained in the previous chapter, but allowing for different intensification factors (from the dictionaries), and making more explicit the inheritance between models sent2 and sent1. Figure 5.3 shows the intensifiers are inherited from the INTENSIFIER node in the \textit{Galadriel} base model and modelled with their own factor values.
Figure 5.2: Model sent2 for intensifiers inheriting from model sent1 with extended rule

Figure 5.3: The intensifiers are structured in a hierarchy in the *Galadriel* base model

Figure 5.2 shows that *good* overrides the default score value inherited from sent1. Additionally, the sent2 model has a rule that recalculates the score value of the word by multiplying the factor value of its previous word, if its previous word is an
intensifier, as follows:

\[
\text{if } \text{prev } \text{Word} = \text{INTENSIFIER word} \text{ then}
\]

\[
\text{Word}_{\text{mag}} = \text{Word}^2_{\text{score}} \times \text{prev } \text{Word}_{\text{fac}}
\]

\text{else}

\[
\text{Word}_{\text{score}} = \text{Word}^3_{\text{score}}
\]

\text{end if}

Feature value of a word is denoted by Word^a_y, where a is model (sent1, sent2, etc.) and y is feature (magnitude, factor, score, etc.).

---

Feature values and rules are inherited from sent1

<sent2> == <here sent1>
<sent2 total> == Eval: <<here sent2 score> + <here sent2 prev sent2 total> .>

<sent2 score> == < IFEQ:< <here sent1 prev1 sent1 type > intensifier THEN case intensifier ELSE case default .> >

<case intensifier> == Eval:< <here sent1 score> * <here sent1 prev sent1 factor>.>
<case default> == <here sent1 score>

A new rule is added to sent2

---

Figure 5.4: The Galadriel code for the model sent2 with additional rules to handle intensification

Then the score of each word accumulated into a total score and the final result is the total value of the final word. Figure 5.4 shows the Galadriel code for the model sent2.

5.1.1.3 Model sent3: Negation

Two methods have been proposed for dealing with negators. They are the switch negation method, where the polarity of the lexical item next to the negator will be switched, and the shift negation method, where the SO value of a word which needs to be negated is shifted towards the opposite polarity by a fixed amount. Negation words include not, never, no, nobody., and I grouped these as NEGATORs. Similar to intensifiers, negators do not have SO values themselves and so are categorised as neutral. Taboada et al. (2011) argue that the switch negation does not work in certain cases. I tried modelling both Taboada et al.’s negation methods in Galadriel. I also used their constant value 4 to recalculate the score for the shift negation.

In this model, the new feature neg-context is used for every word in the document. As I explained in chapter 4, the feature neg-context takes the value yes or no. As
Figure 5.5: Model sent3 for negation: the score is adjusted by neg-context

a default value, all negation words take the neg-context value yes and any word in a clause with a negator is assigned the neg-context value yes, otherwise no.

Moreover, Taboada et al. (2011) defined any negator as a word negating the opinion word within the same clause. In order to identify a clause or sentence, a list of end punctuation such as '.', '!', etc., was created. This allows the identification of clauses and sentences in a document. Then the model allows each word to take its previous word’s neg-context value, until it gets to the end punctuation. I discussed punctuation in the previous chapter, and categorised it as BOUNDARY. In that way, any words present after a negator within the clause take the neg-context value yes.

\[
\text{if the Word’s type is NEGATION then} \\
\text{Word}_{\text{neg-context}} = \text{Word}_{\text{base}}^{\text{neg-context}} \\
\text{else} \\
\text{if prev Word = BOUNDARY word then} \\
\text{Word}_{\text{neg-context}} = \text{no} \\
\text{else} \\
\text{Word}_{\text{neg-context}} = \text{prev Word}_{\text{neg-context}} \\
\text{end if} \\
\text{end if}
\]

Then the negation rule(shift) is applied to the words that have the neg-context
value of \textit{yes}. Finally, following SO-CAL, the shift negation rule is applied to words which have a \texttt{neg-context} value of \textit{yes}, as follows:

\begin{verbatim}
if Word\_neg-context = yes then

if Word\_score > 0 then
    Word\_score = Word\_score - 4;
else

if Word\_score < 0 then
    Word\_score = Word\_score + 4;
else
    Word\_score = Word\_score
end if
end if
end if
\end{verbatim}

\begin{figure}[h]
\centering
\begin{verbatim}
<sent3> == <here sent2>
<sent3 total> == Eval:< <here sent3 score> + <here sent3 prev sent3 total> .>

<sent3 neg-context> == < IFEQ:< <here sent3 type > > boundary
    THEN case skip-found ELSE case negation-context . > >
    <case skip-found> == no
    <case neg-found> == yes
<case negation-context> == < IFEQ:< <here sent3 type > > negation
    THEN case neg-found ELSE test negation-context . > >
    <test negation-context> == <here sent2 prev sent2 neg-context>

<sent3 score> == < IFEQ:< <here sent2 prev sent2 word > > negation
    THEN test positive ELSE case no-negator . > >
    <case no-negator> == <here sent1 score>
    <test positive> == < IFEQ:< Compare:< <here sent2 score> > 0> more
        THEN case positive ELSE test negative . > >
    <test negative> == < IFEQ:< Compare:< <here sent2 score> > 0> less
        THEN case negative ELSE case no-negator. > >
    <case positive> == Eval:< <here sent2 score> - 4.>
    <case negative> == Eval:< <here sent2 score> + 4.>
\end{verbatim}
\caption{The \textit{Galadriel} code for the model sent3 with the additional rule for negation}
\end{figure}

As figure 5.6 shows, model sent3 is inherited from the sent2 model, and sent3 calculates the total value of each word. Figure 5.6 shows the \textit{Galadriel} code for the sent3 model. Similarly, the switch negation feature was modelled by switching the \texttt{polarity} of the word, if its \texttt{neg-context} value is \textit{yes}. 
5.1.1.4 Model sent4: Irrealis Blocking

In *Galadriel*, Taboada et al.’s list of irrealis markers is categorized under a hierarchical lexical node called MARK. To model SO-CAL’s irrealis blocking feature in *Galadriel*, a new feature called *block-context* with possible values *yes* or *no* was introduced. Similar to model sent3, the *block-context* feature also uses end punctuation words to assign its own value, as irrealis blocking applies only within a clause or sentence. The sent4 model handles irrealis blocking, which is inherited from sent3 (see 5.7). The following algorithm is used to handle the irrealis blocking feature in sent4:

```python
if Word{block-context} = yes then
    Word{score} = 0
else
    Word{score} = Word{score-1}
end if
```

Figure 5.7: Model sent4: *block-context* changes sentiment scores to 0
<sent4> == <here sent3>
<sent4 total> == Eval:< <here sent4 score> + <here sent4 prev sent4 total> .>

<sent4 block-context> == < IFEQ:< <here sent3 type.> boundary
    THEN case skip-found4 ELSE test irrealis-blocking .> >
    <case skip-found4> == no
    <case block-found> == yes
    <test irrealis-blocking> == < IFEQ:< <here sent3 type.> mark
    THEN case block-found ELSE test block-context .> >
    <test block-context> == <here sent3 prev sent3 block-context>

<sent4 score> == < IFEQ:< <here sent4 block-context .> yes
    THEN case irrealis-blocking ELSE case default4 .> >
    <case irrealis-blocking> == 0
    <case default4> == <here sent3 score>

(a) The Galadriel code for calculating score in model 4 by considering block-context

<sent4a> == <here sent4a>
<sent4a total> == Eval:< <here sent4a score> + <here sent4a prev sent4a total> .>
    <sent4a ques-context> == < IFEQ:< <here sent4 type.> boundary
    THEN case skip-found4a ELSE test det-found .> >
    <case skip-found4a> == no
    <case det-found> == no
    <case quest-found> == yes
    <test det-found> == < IFEQ:< <here sent4 prev sent4 type.> determiner
    THEN case det-found ELSE test ques-blocking .> >
    <test ques-blocking> == < IFEQ:< <here sent4 next sent4 word.> \?
    THEN case quest-found ELSE test question-context .> >
    <test question-context> == <here sent4a next sent4a ques-context>

<sent4a score> == < IFEQ:< <here sent4a ques-context .> yes
    THEN case question-blocking ELSE case default4a .> >
    <case question-blocking> == 0
    <case default4a> == <here sent4 score>

(b) The Galadriel code for handling the question context feature

Figure 5.8: The Galadriel code for calculating score model sent4 by considering irrealis blocking

In addition, the ques-context feature is used to decide whether the clause/sentence is a question. Then, if any determiners are found within the clause/sentence, irrealis blocking is ignored (see figure 5.7). Figure 5.8 shows the Galadriel code for model sent4.

Model sent5 and Model sent6: Text-Level Features

Taboada et al. (2011) believe lexicon-based sentiment classifiers generally favour positive language statements and so previous sentiment research shows a positive
bias. Moreover, they state that the repetition of a sentiment word found in a sentence shows sentiment depending on how many times the sentiment word is present in the sentence.

**Figure 5.9:** Model sent5: changes sentiment score of the word, dependent on its word count

SO-CAL may show strong positive sentiment, for example as seen in figure 5.9, due to the repetition of the word excellent. However, Taboada et al. (2011) suggest that the number of appearances of a sentiment word in a sentence should not decide its overall sentiment intensity. In order to overcome this problem, firstly SO-CAL increased the final SO value of any negative expression by 50%. Secondly, they decreased the weight of words which appear more often in the document. In this way, they decided to override the SO value of the \( n^{th} \) appearance of a word with \( 1/n \) of its full SO value.

To model SO-CAL’s feature for weight of repeated words in *Galadriel*, a new feature called **count $\text{word}$** was introduced, where ‘$\text{word}$’ is a DATR variable, so this definition works for different actual words, for instance <count excellent>, <count horrid>. This feature allows us to count how many times a word is present in a document. Thus the sentiment score of the word ($\text{word}$) is divided by **count $\text{word}$** to produce the final score for the word (see figure 5.9). To model negation weighting, first, the system decides whether the overall sentiment is negative. If so, the total score is increased by 50% (see figure 5.10). I introduced a new feature called **weighted-score**, which is defined for every word, that recalculates the total score by increasing it by 50% if the total score is a negative value. Finally, the final score for the document gets the total score of the last word of the document, which
is identified by testing the end punctuation word and the word next to it.

Figure 5.10: Model sent6: changes the total score by reducing the total score by 50% if it is negative

Figure 5.11: The *Galadriel* code for the sent5 and sent6 models
5.2 Evaluation of SO-CAL features

I collected the whole dataset and the dictionary used by SO-CAL. SO-CAL’s dictionary contains a list of words (adjectives, adverbs, nouns and verbs) with their SO (semantic orientation) values (between −5 and +5). In addition, it has a list of intensifiers with their values in factors (with a plus or minus sign). I tested SO-CAL features in Galadriel using two datasets, which were based on those used in Taboada et al. (2011). The datasets are:

- **Epinions**: Taboada et al. (2011) used a collection of 400 texts used by Taboada and Grieve (2004) which contain 50 reviews each of books, cars, computers, cookware, hotels, movies, music and phones.

- **Movies**: 1900 texts from the polarity dataset Pang and Lee (2004).

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Only adjectives</th>
<th>All words</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SO-CAL</td>
<td>Galadriel</td>
</tr>
<tr>
<td>Epinions</td>
<td>72.25</td>
<td>75.50</td>
</tr>
<tr>
<td>Movies</td>
<td>76.63</td>
<td>76.75</td>
</tr>
</tbody>
</table>

Table 5.2: Performance of SO-CAL and Galadriel models for only adjective and all words

<table>
<thead>
<tr>
<th>Features</th>
<th>SO-CAL</th>
<th>Galadriel</th>
<th>Features</th>
<th>SO-CAL</th>
<th>Galadriel</th>
</tr>
</thead>
<tbody>
<tr>
<td>simple</td>
<td>65.50</td>
<td>68.05</td>
<td>Models</td>
<td>sent1</td>
<td>66.25</td>
</tr>
<tr>
<td>negation</td>
<td>67.75</td>
<td>70.10</td>
<td>sent1a</td>
<td>73.25</td>
<td>75.31</td>
</tr>
<tr>
<td>neg+intensifiers</td>
<td>69.25</td>
<td>73.47</td>
<td>sent2</td>
<td>75.75</td>
<td>76.42</td>
</tr>
<tr>
<td>neg+inten+irrealis</td>
<td>71.00</td>
<td>74.95</td>
<td>sent4</td>
<td>79.50</td>
<td>78.41</td>
</tr>
<tr>
<td>neg+inten+irr+neg weight</td>
<td>81.50</td>
<td>75.08</td>
<td>sent4a</td>
<td>83.25</td>
<td>79.96</td>
</tr>
<tr>
<td>neg+inten+irr+neg w+rep w</td>
<td>80.25</td>
<td>76.37</td>
<td>sent6</td>
<td>83.25</td>
<td>80.16</td>
</tr>
<tr>
<td>neg(swi)+inten +ir + neg w +rep w</td>
<td>80.00</td>
<td>75.57</td>
<td>sent6aa</td>
<td>80.25</td>
<td>78.45</td>
</tr>
</tbody>
</table>

Table 5.3: Comparison of the performance of SO-CAL features and Galadriel models (all words)

The Galadriel system was tested in several configurations, simulating SO-CAL’s ‘only adjectives’ and ‘all words’ settings (including sentiments for adverbs, nouns and verbs), and for all six Galadriel models (sent1 to sent6). Table 5.2 shows the performances of SO-CAL and Galadriel with adjectives and all words in sent1 and sent6. Table 5.3 provides a comparison of the performance of SO-CAL (all words) with different features and different models of Galadriel (all words). To compare...
SO-CAL features and *Galadriel* models directly, I made slight changes to the *Galadriel* models. To compare *Galadriel* with SO-CAL’s negation feature, I re-modelled *Galadriel*’s inheritance structure by making *Galadriel* negation model inherit from the sent1 model, named ‘sent1a’. Similarly, the *Galadriel* sent5 model contains the models of the SO-CAL features, negation, intensification, irrealis blocking and repetition weight, which are inherited one from the other. So, I insert the ‘negation weight’ model, which inherits from the irrealis blocking model (*Galadriel* sent5) and is named ‘sent4a’. Then I ran the *Galadriel* models sent1, sent1a, sent2, sent3, sent4, sent4a and sent6 with the datasets separately, and their performance is shown in Table 5.3.

Moreover, I also computed f-scores for positive and negative reviews of the Epinions datasets separately. Table 5.4 indicates the comparison of the performance of SO-CAL and *Galadriel* across review types and on positive and negative reviews. F-score and accuracy of *Galadriel* for positive and negative reviews give 0.84, 0.82 and 0.83 respectively. Moreover, *Galadriel* shows 80.16% of accuracy for overall movie reviews.

### 5.3 The Opinion Observer System

[Ding et al. (2008)](Ding et al. 2008) proposed a method called ‘A Holistic Lexicon-Based Approach to Opinion Mining’ and built a system called *Opinion Observer* (OO). This work mainly focused on two areas: (1) opinion words which are context dependent, (2) aggregating multiple opinions in the same sentence. Moreover, this approach is feature-/aspect-based sentiment analysis. The method aimed to extract the sentiment towards each component/aspect in a product. For example, a cellular phone
company may be interested in analysing customers’ reviews of all the components (speaker, camera, battery, etc.) of their phones. To complete the task, they defined a model which identified a set of features, $F = f_1, f_2, f_3, ..., f_n$ and a set of words/phrases, $w_i$, which can express each feature $f_i$. This set of words/phrases are synonym sets $W = W_1, W_2, W_3, ..., W_n$ for the $n$ features. The opinion holder chooses a word or phrase from the set $W_k$ to describe the feature $f_k$, and expresses a positive, negative or neutral opinion on the particular feature.

In this model, three main problems were introduced:

Problem 1: Both $F$ and $W$ are unknown.
Problem 2: $F$ is known, but $W$ is unknown.
Problem 3: $W$ is known (then $F$ is also known)

Identifying the feature is the most important task for any problems. However, the authors’ previous work showed how to extract object features from the given reviews. Therefore, this work was done by assuming the feature is given, and it thus focused on problem 3. This holistic work (Ding et al., 2008) only aims to analyse polarity of sentiment towards a given feature, not its intensity.

5.3.0.5 Opinion Lexicons

Opinion lexicons are sets of words and phrases that are used to express the sentiment of a statement. The same set of opinion lexicons used in Hu and Liu (2004) were used in this work. In addition, the researchers added some more opinion verbs and nouns, and a list of context-dependent words. To make use of different parts of speech, they used the NLProcessor linguistic parser for POS tagging.

5.3.0.6 Aggregating Opinions for a Feature

Ding et al. (2008) work mainly focuses on finding an opinion orientation (positive, negative or neutral) expressed in regards to a given product feature in a statement/review. In order to decide the opinion orientation, the system computes a semantic orientation score for the feature. Each of the positive and negative words is assigned semantic orientation scores of +1 and −1, respectively. A semantic score of 0 is assigned to neutral words. The semantic orientation for the feature is calculated by aggregating the semantic orientation scores of all the words present in the review. If this final score is positive, then the opinion (orientation) in regards to the feature is positive. On the other hand, if the final score is negative, then the

opinion of the feature is negative. This approach is slightly different from SO-CAL because it relates to a particular feature. Apart from that, this calculation is the same basic one we saw before, but the difference is it only considers signs. Also, the system considers the distance between the feature and the opinion words, as the opinion words that are far away from the feature may not modify the feature. The system uses the following equation to aggregate all scores to get the final semantic orientation score for the feature:

\[
\text{Score towards the feature } f = \sum w_i \text{SO} / \text{dis}(w_i, f) \tag{5.1}
\]

where,

- \( w_i \) is an opinion word
- \( w_i \text{SO} \) is the semantic score of the word \( w_i \)
- \( f \) is the given feature
- \( \text{dis}(w_i, f) \) is the distance between the feature \( f \) and the word \( w_i \)

### 5.3.0.7 Negation

A negation word usually reverses the opinion orientation in the same way as the switch negation method in SO-CAL. Thus, the system detects any negation words in the sentence and then substitutes -1 for positive words and +1 for negative words. Non-negation terms containing negation words/phrases such as *not just* or *not only* are also identified, and the semantic orientation of their negation words is overwritten by +1.

### 5.3.0.8 Handling Context-Dependent Opinions

The most significant feature of this holistic approach is that the system is specially designed to focus on context-dependent opinion words. Context-dependent opinion words are non-opinion words, but they express opinion depending on their context. Usually, they are adjectives. Three linguistics conventions were used to deal with context-dependent words that are employed in reviews of the same product:

1. **Intra-sentence conjunction rule**:  
   The system uses conjunction words, *and* and *but*, to decide the opinion orientation of context-dependent opinion words. If two clauses are joined with *and*, those two clauses express one orientation opinion. On the other hand, if two clauses are joined with *but*, then those two clauses should represent the opposite opinion orientation:
(a) ‘This camera takes great pictures and has a long battery life.’
(b) ‘This camera takes great pictures but has a short battery life.’

The Opinion Observer system decides the semantic orientation of the context-dependent opinion words long and short in the above sentences, using the conjunction words and and but. In example (a), it can be seen that great is positive for picture. Thus, long in the following clause has been assigned a positive opinion orientation for the feature battery life, as the clauses are joined with and. In contrast, but in example (b) changes the direction of the orientation.

2. **Pseudo intra-sentence conjunction rule:**

The pseudo intra-sentence conjunction rule is used to detect the semantic orientation of context-dependent words, where the conjunction and has not been used explicitly. Consider the following example:

‘The camera has a long battery life, which is great.’

In the above example, there is no clear idea of whether long has been used to express a positive or negative opinion for the feature battery life. However, this has been decided by looking at the following clause, which is great. The word great is a positive word. Hence, the system assigns +1 to the semantic orientation score long for battery life.

3. **Inter-sentence conjunction rule:**

This rule is used when context-dependent words cannot be decided by the above two rules. In this rule, the context of a sentence/clause is used to decide the next sentence/clause. In other words, the intra-sentence conjunction (and) rule has been extended to the neighbouring sentence. Example:

‘The picture quality is amazing. The battery life is long.’

The semantic orientation of the word amazing is positive for picture quality, so the system decides that the context opinion word long expresses a positive view on battery life too.

5.3.0.9 **Additional Considerations**

The above features were used in the Opinion Observer system to handle the opinion mining task by Ding et al. (2008), while taking account of the following considerations:
• Synonyms of positive (or negative) words within a context are considered to be positive. On the other hand, antonyms are considered to be negative (or positive) within the same context.

• The negation rule is applied to any sentence if the word *too* is presents before a context-dependent opinion word. Example:

> ‘The camera is too small.’

*small* in the above example does not express any opinion on *camera*. However, *too* indicates a negative view. Therefore, the negation rule is applied to any context dependent words which are preceded by *too*.

• Sometimes, adjectives (opinion words) can be represented as feature indicators. In such cases, equation 5.1 is not used to calculate the semantic orientation of product features, and the semantic orientation of the opinion word is directly assigned for the score on the product. For example:

> ‘This camera is reliable.’

In this example, *reliable* can be a product feature (*reliability*). Thus, the above sentence expresses the positive/negative opinion on the product feature *reliability*, depending on whether *reliable* is positive or negative, which can be found out from the opinion dictionary (for opinion words) or using the context dependent opinion rules (for context-dependent opinion words).

Table 5.5 shows the summary of OO features that have been used to calculate semantic orientation of a text towards a given a feature. Moreover Diagram 5.12 summarises the OO’ s sentiment analysis method.

<table>
<thead>
<tr>
<th>OO features for the calculation</th>
<th>Calculation of SO value</th>
</tr>
</thead>
<tbody>
<tr>
<td>for any word $w_i$</td>
<td>$w_i SO$</td>
</tr>
<tr>
<td>For negation word</td>
<td>$-w_i SO$</td>
</tr>
<tr>
<td>For context dependent words</td>
<td>Use linguistics conventions</td>
</tr>
<tr>
<td>Score towards the feature</td>
<td>$f = \sum w_i SO / \text{dis}(w_i, f)$</td>
</tr>
</tbody>
</table>

Table 5.5: Summary table of OO features and their calculation
POS Tagging

Is it an opinion word?

YES

Find the semantic orientation score (pos/neg/neu) (using option lexicons)

NO

Is it an adjective?

YES

Is the adjective followed by 'too'?

YES

Apply context dependent opinion rules

NO

Apply the negation rule

Assign the semantic orientation score $w_{i}.SO$ (+1 or -1) for the word

NO

Check, if there are any negation words in the sentence?

YES

Apply context dependent opinion rules

NO

The semantic orientation score on the product feature $f_i = w_{i}.SO/dis(w_i,f)$

YES

The semantic orientation score on the product feature $F = \sum w_{i}.SO/dis(w_i,f)$

NO

Total semantic orientation score on the product feature $F = \sum w_{i}.SO/dis(w_i,f)$

$dis(w_i,f) = \text{distant between } i^{th} \text{ word and the feature}$

Figure 5.12: A summary diagram of the Opinion Observer method
5.3.0.10 Evaluation of Opinion Observer

Figure 5.12 provides a summary of the Opinion Observer system’s method. The performance of the system was tested using an empirical evaluation method. Ding et al. (2008) examined the full Opinion Observer by comparing it with three other techniques: (1) the Opinion Observer without using equation 5.1, (2) the Opinion Observer without context-dependency handling and (3) their previous system, FBS (Hu and Liu, 2004). Experiments using customer reviews of 8 products, including one DVD player, two digital cameras, one MP3 player, two cellular phones, one router and one piece of antivirus software, were carried out in order to evaluate the system. The reviews of the first five products were taken from a benchmark dataset, and the reviews of the rest of the products were human-annotated. Firstly, POS tags for the datasets were generated using the NLP processor 3, then the system was applied to the datasets to get the final results on opinion orientation towards product features. The performance was measured using the standard evaluation measures, precision, recall and f-score. The results were compared across the three techniques. An average f-score of 0.90 was produced by this method, which is better than the other methods.

5.3.1 Modelling OO in Galadriel

Similar to modelling SO-CAL in Galadriel, I modelled OO in Galadriel. We studied Ding et al. (2008)’s techniques and methods and aimed to model Galadriel using an inheritance-based structure. Unlike SO-CAL, OO deals only with polarity or semantic orientation, but it has an extra task, which is identifying the product feature (or aspect) in a sentence. OO were modelled in Galadriel by making an assumption that the product features are given. This section outlines the steps of the modelling process of both my methods. I used the same dictionary (opinion lexicon) and datasets 4 as Ding et al. (2008) used in their work. The dataset contains customer reviews of nine products. The reviews are short sentences. I also collected their sentiment lexicon dictionary, which has a set of positive and negative words with scores of +1 and -1, respectively. As I assumed the features/aspects had already been given, the task was to extract the sentiment towards the given features or aspects. In future, I use the term ‘aspect’ for the product ‘feature’, in order to avoid any terminological confusion.

3NLPProcessor – Text Analysis Toolkit.2000-www.infogistics.com/textanalysis
4https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
As the first step of the modelling process, I built Galadriel’s sentiment lexicon using Ding et al. (2008)’s sentiment lexicon. Ding et al. (2008)’s OO system was mainly designed to handle context-dependent words by exploiting global information. The current version of Galadriel is not able to use global information directly. Therefore, I divided the process into two steps, collecting a list of context-dependent words and annotating them with their SO value using Ding et al. (2008)’s linguistic conventions, before plugging them into the Galadriel lexicon. A list of neutral adjectives (context-dependent words, e.g. small, big, short, etc.) were compiled from a web dictionary.\footnote{http://www.gingersoftware.com/content/grammar-rules/adjectives/lists-of-adjectives/}

Then I used a part of the annotated customer review dataset and extracted SO values of the collected context-dependent words using the rules of the linguistic convention and ‘too’ rules manually. In this way, there are some downsides to this method. I annotated context-dependent words with an SO value depending on the reviews, but not on the aspects. For example, assume on a phone review, I found the context-dependent word big is negative for the aspect size, and long is positive for battery. So I assume for all aspects of the product phone, the SO value of big is -1 and long is +1. I modelled the collected context-dependent words under the nodes...
CON-POSITIVE and CON-NEGATIVE, which are inherited from the CONTEXT node, as shown in figure 5.13.

Moreover, the targeted aspect (product feature) words and their indicators were collected, and modelled them under a node called ASPECT, which is inherited from the node NEUTRAL, as shown in figure 5.13 To model OO features in Galadriel, I built models such as adding up the SO value of opinion words towards the given aspects, negation, handling context-dependency and aggregating SO value for the aspects, using the equations inherited one from the other, and named the sent1, sent2, sent3 and sent4.

5.3.1.2 Model sent1: Aggregating SO towards the given feature/aspect

I have a lexicon for opinion words (+1 for positive and -1 for negative words) and context-dependent opinion words (annotated with the help of the other reviews using linguistic conventions) with their SO values. To get the total score for each targeted aspect in a given sentence, first, this model checks if the targeted aspect (let’s say $aspect$) is present in the document/sentence.

\begin{itemize}
  \item **Aspects words in the document:** First, the sent1 model identifies the targeted aspect present in the text/document. For this task, I added four additional Galadriel feature:value pairs to each word lexical agent:
    \begin{itemize}
      \item **found ASPECT\textsubscript{i}:**
        Every word lexical agent has value of \texttt{found ASPECT\textsubscript{i}} with respect to all given aspects, where $i= 1, 2, 3, ...$. Every lexical item in a sentence can have \texttt{found-ASPECT\textsubscript{i}} value either \texttt{true} or \texttt{fail}. For instance, The feature \texttt{found ASPECT\textsubscript{1}} of a Word\textsubscript{1} in a sentence with the value \texttt{true} indicates that the aspect ASPECT\textsubscript{1} is present in the specific sentence. In order to assign \texttt{found ASPECT\textsubscript{i}} value for each word, I introduce another two sub features, \texttt{found-right ASPECT\textsubscript{i}} and \texttt{found-left ASPECT\textsubscript{i}}, which are explained in the next section in detail. The default values of \texttt{Word\textsubscript{found-ASPECT\textsubscript{i}}}, \texttt{Word\textsubscript{found-right ASPECT\textsubscript{i}}} and \texttt{Word\textsubscript{found-left ASPECT\textsubscript{i}}} are set to \texttt{fail}.
      \item **score-ASPECT\textsubscript{i}:**
        This value of each word indicates, the word’s sentiment score towards ASPECT\textsubscript{i}. The default value of each word ( \texttt{Word\textsubscript{score ASPECT\textsubscript{i}}}) is 0.
      \item **total-ASPECT\textsubscript{i}:**
        Similar to the total feature, \texttt{total ASPECT\textsubscript{i}} gives the total sentiment score of the document towards the aspect ASPECT\textsubscript{i}.
    \end{itemize}
\end{itemize}
The sent1 model has rules to get the right values for the newly introduced features outlined above. I set 0 as default values for $Word_{total}^{ASPECT_i}$, $Word_{senti-score}^{ASPECT_i}$.

The rules for getting the right value for found ASPECT$i$: This model identifies the targeted aspects within a sentence, hence, sent1 assigns appropriate values (either true or fail) to the found ASPECT$i$ feature for each word in the sentence.

It is easy to assign found ASPECT$i$ values to any ASPECT$i$ term or its indicators in the sentence. However, it is not straightforward to assign found ASPECT$i$ values for other words in the sentence. Let’s take an example review from the phone domain, with targeted aspects, SCREEN, BATTERY:

‘The screen is super high quality.’

Each word of the above sentence is referring the aspect SCREEN, as only SCREEN is present in the sentence. None of the words are referring the aspect BATTERY. The value for found ASPECT$i$ can be easily assigned as follows:

\[
\begin{array}{ccccccc}
\text{The screen is super high quality .} \\
\end{array}
\]
The word ‘screen’ in the above example easily takes the value of true for the feature found SCREEN, because screen is an aspect (SERVICE) word or its indicator. However, the other words cannot find their found SERVICE value. In order to overcome this issue, found ASPECT<sub>i</sub> is divided into two features: found-right ASPECT<sub>i</sub> and found-left ASPECT<sub>i</sub>. Similarly those two features can take either true or fail. The feature found-right ASPECT<sub>i</sub> means that the words present the right side of the ASPECT<sub>i</sub>. The value true is assigned for the feature found-right ASPECT<sub>i</sub> for any word that indicates its presence on the right side of ASPECT<sub>i</sub> within the same sentence.

To assign the value found-right ASPECT<sub>i</sub>, for each word;

\[
\text{if } \text{Word} = \text{ASPECT}_i \text{ word node then}
\]
\[
\text{Word}^{\text{ASPECT}_i} = \text{true}
\]
\[
\text{else}
\]
\[
\text{Word}^{\text{ASPECT}_i} = \text{next } \text{Word}^{\text{ASPECT}_i}
\]

\[
\text{end if}
\]

Similarly, to assign the value found-left ASPECT<sub>i</sub>, for each word;

\[
\text{if } \text{Word} = \text{ASPECT}_i \text{ word node then}
\]
\[
\text{Word}^{\text{ASPECT}_i} = \text{true}
\]
\[
\text{else}
\]
\[
\text{Word}^{\text{ASPECT}_i} = \text{prev } \text{Word}^{\text{ASPECT}_i}
\]

\[
\text{end if}
\]

Now using the above feature values, the found ASPECT<sub>i</sub> value for each word can be assigned as follows:

For every word,

\[
\text{if } \text{Word}^{\text{ASPECT}_i} = \text{true or } \text{Word}^{\text{ASPECT}_i} = \text{true then}
\]
\[
\text{Word}^{\text{ASPECT}_i} = \text{true}
\]
\[
\text{else}
\]
\[
\text{Word}^{\text{ASPECT}_i} = \text{fail}
\]

\[
\text{end if}
\]

Let’s consider the example;

‘The screen is super high quality.’

Figure[5.14] demonstrates getting the right values of found SCREEN and found BATTERY using the above rules. Hence, the system knows all the words present
in the sentence are referring to the aspect SCREEN as each word has the value of \( \text{Word}_{\text{found}}^{\text{SCREEN}} \) as \( \text{true}(T) \), whereas the feature \( \text{Word}_{\text{found}}^{\text{BATTERY}} \) value of each words is \( \text{fail}(F) \). This means that the words of the above sentence do not refer to the aspect FOOD. This is modelled in the Galadriel model sent1 as shown in figure 5.15.

Assigning score-ASPECT\( i \) and calculating the total-ASPECT\( i \) values:
Once the model has identified the targeted aspects(ASPECT\( i \)) in the text, the model assigns the SO of the lexical item to the targeted aspect score-ASPECT\( i \) which is found in the document. Then the model aggregates the total score of each aspect by adding its previous aspect score. Assume, the targeted aspects are SIZE,SCREEN and BATTERY, and consider the following example:

‘Its speaker is good and the battery life is fine.’
Figure 5.16: sent1 model: assigning SO values to the targeted features calculates the total feature score

Figure 5.16 shows that the targeted aspects, SCREEN, BATTERY and SIZE, are getting the scores of lexical items and being aggregated to the total.

5.3.1.3 Model sent2: Negation

Negation words or phrases usually reverse the polarity/semantic orientation of a sentence. Negation words traditionally include *no*, *not*, *never*, etc. Ding et al. (2008) also considered negation verbs such as *stop*, *quit*, *cease*, etc. Logically, the semantic orientation of a word which comes after a negator is reversed.

Similar to what I introduced in SO-CAL for negation rules, for this model too I used a Galadriel feature to mark negation context, `neg-context(yes or no)`, for each word. Any word within a negation context (`neg-context = yes`) switches its semantic orientation (positive to negative or negative to positive) by itself. I set the default `neg-context = no` (which means the particular word does not have a negation context and the semantic orientation of the word remains the same). So the initial `neg-context` at the start of the document is `no` . The default behaviour for a word is that its `neg-context` is the same as the previous word. But negators set it to `yes` . Moreover, similar to the modelling of SO-CAL, I use BOUNDARY items to detect clauses; at the end of the clause, `neg-context` is set back to `no`. 

103
In addition, Ding et al. (2008) also discussed a special case, non-negation containing negation words, such as *not only*, which are also modelled in *Galadriel*, as its ELF mechanism allowed each lexical item to access the information of its neighbouring item. The model checks if *only* is present after the negator *not*, then the feature, **neg-context**, of the word *not* is changed to *no*.

As the rules of the sent1 model are inherited, the sent2 model calculates the total SO value of each target feature, as shown in figure 5.17 for the following example sentence:

> ‘*Its speaker is not good but the battery life is fine*.’

In the above sentence, the SO value of *good* is +1 and its neg-context value is *yes*. So the SO value is switched to -1. Then the BOUNDARY word *but* changes the neg-context value of following words take their previous neg-context value.

### 5.3.1.4 Model sent3: Handling Context Dependency

I modelled OO’s context dependency feature in sent3. As I explained, I collected opinion-dependent words with their SO values (using intra-sentence and Pseudo intra-sentence conjunction rules) and added them to the *Galadriel* lexicon. For any words that not are assigned an SO value, sent3 uses the ‘too’ rule and Ding et al. (2008)’s Inter-sentence conjunction rule to assign a **score** value to CONTEXT.
words using the below algorithm and figure 5.18 shows the calculation of SO value for context words. This allows us to use local information to assign an SO value to context-dependent words, such as long, short, big, small, etc. I used CONTEXT nodes to identify those words.

For context word,

if prev Word = too then
    \( \text{Word}^2 \text{score} = -1 \)
else

    if the sentence previously contains however or but then
    \( \text{Word}^3 \text{score} = -1 \times \text{SO of opinion word within the sentence} \)
else
    \( \text{Word}^3 \text{score} = \text{SO of opinion word within the sentence} \)
end if
end if

![Diagram of aspect score calculation](image)

Figure 5.18: The sent3 model handles context-dependent words using local information.

I also introduced the following Galadriel features:

- First, the model detects if the context-dependent word is found next to too, then the too rule is applied to the model.

- sentence-score and found Inter: These two Galadriel features are used to assign the score value to a CONTEXT word, when opinion words are found in the clause before the clause that has the CONTEXT word and use however.
or or to join the clauses. Consider the following example:

'It's screen is good but the battery life is short'

**found Inter** has the default value *fail*, while it takes *true* when the intersentence conjunction rule is satisfied. The **sentence-score** feature value of a word can take either +1 or -1, according to its previous word and its **found Inter** value. The *Galadriel* code is shown in figure 5.19.

```
<sent3 sentence-score> == < IFEQ:< <here sent2 score.> 0
                      THEN test-sentence ELSE case-sentence>
                     <test-sentence> == <here sent3 prev sent3 sentence-score>
                      <case-sentence> == <here sent2 score>

<sent3 found inter> == < IFEQ:< <here sent3 word .> however
                        THEN case-inter ELSE test-but .> >
                     <test-but> == < IFEQ:< <here sent3 word .> but
                       THEN case-inter ELSE inter-case .> >
                       <inter-case> == <here sent3 prev sent3 found inter>
                      <case-inter> == **true**
                       <inter-fail> == **false**

<sent3 score> == < IFEQ:< <here sent3 type .> context
                 THEN test-too ELSE inter-rule .> >
                 <test-too> == < IFEQ:< <here sent3 prev sent3 word.> too
                  THEN case too ELSE inter-rule .> >
                 <case too> == -1
                 <inter-rule> == < IF:< <sent3 found inter .>
                    THEN case-inter ELSE default-inter .> >
                 <case-inter> == Eval:< <here sent3 sentence-score> * -1 .>
                 <default-inter> == Eval:< <here sent3 sentence-score > .>
```

Figure 5.19: The *Galadriel* code for sent3 model

### 5.3.1.5 Model sent4: Distance Between Targeted Feature and Opinion Word

In the OO system, the distance between a targeted aspect and a sentiment word affects the final sentiment score of the targeted aspect. Consider the following example:

'It has a great screen with a horrid battery.'

Similarly, assume the given targeted aspects are SCREEN, BATTERY and SIZE. The *Galadriel* sent2 model assigns 0 for score-SIZE because the *size* is not mentioned in the sentence. It produces the semantic orientation score 0 for SCREEN and BATTERY because, although they are mentioned, the total sentiment score for each is +1 –1, which is 0. However, the sentence expresses a positive semantic orientation
on SCREEN and a negative semantic orientation on BATTERY. If we consider the distance between the opinion words and the targeted features, the positive word (great) is closer to the targeted aspect phone, and the negative word (horrid) is closer to the targeted aspect battery. In the original work, they used the following equation to compute the orientation of an aspect:

\[
\text{Score towards the feature, Aspect}_i = \sum w_i \cdot \text{SO} / \text{dis}(w_i, f)
\]

Opinion words, which are far away from the targeted aspect Aspect$_i$, are given low weights by using this multiplicative inverse in the formula because such opinion words may not express any opinion on the aspect.

The Galadriel sent4 model is inherited from sent3 and is designed to calculate total targeted aspect score by taking account of the distance between the opinion word and the aspect.

To model this feature in Galadriel, I introduced a new feature called \textbf{dis-Aspect} which allows the calculation of the distance between a word and all the targeted features (Aspect$_i$) present in the sentence using Ding et al. (2008)'s equation, given above. To calculate the distance, each lexical item in the document/sentence follows the following steps:

Figure 5.20: The sent4 model calculates \textbf{total Aspect} for each aspect by using the OO equation
• Check if any targeted features ($aspect) are present in the sentence.

• If so, check how far away the word is from each targeted aspect (Aspect$_i$).

If a targeted aspect is not found in the document/sentence, then the system assigns a value of a fixed large number $X$ (in the current implementation, 1000000) for dis $aspect$ which allows the system to decide the final distance between the word and the targeted feature by taking account of the shortest distance. $Galadriel$ sent4 calculates the total targeted aspect score as shown in figure-5.20 and figure-5.21 shows the $Galadriel$ code for modelling the OO equation in sent4.

<sent4> == <here sent3>
<sent4 total $aspect> == Eval:< <here sent4 score $aspect> + <here sent4 prev sent4 total $aspect> .>

<sent4 dist $aspect> == <IFEQ:< <here sent4 word> $aspect
THEN case found ELSE case calculate .> $aspect>
  <case found> == 1
  <case calculate $aspect> == Eval:< <here sent4 next sent4 dist $aspect> + 1 .>
<sent4 score $aspect> == Eval:< <here sent1 score> / <here sent4 dist $aspect> .>

Figure 5.21: The $Galadriel$ code for model the OO equation in sent4

However, this model always produces a value (the fixed large number $X$) for the distance of any targeted aspects which are not found in the sentence, as explained above. Hence, $Galadriel$ returns a value of $X$ for dis screen in the above example. As a result, the sentiment of such sentences towards the unfounded targeted aspect would show a minimal positive or negative value. To overcome this problem, I assume any sentiment scores between $-10^{-5}$ and $+10^{+5}$ are considered to be 0. Thus they are given as neutral. But in the case of similar numbers of positive and negative words found in a sentence, it gives the total aggregation of scores as 0, as shown in the above example.

5.4 Evaluation of OO features

I collected the datasets and opinion lexicon which were used for the evaluation of the OO system, using all eight products. I extracted the product aspects automatically using Sketch-Engine and plugged them in to the $Galadriel$ lexicon manually.

https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
https://old.sketchengine.co.uk/open/
under the node ASPECT. I also added synonyms of the chosen aspect terms to the Galadriel lexicon in order to improve the detection of the targeted aspects.

Similar to the evaluation of the SO-CAL modelling, I compared the OO modelling in the performance of Galadriel performance against the results that were published in [Ding et al. (2008)]’s research paper.

Table 5.6 provides a comparison between the overall performance of OO and Galadriel models.

<table>
<thead>
<tr>
<th>Reviews</th>
<th>Opinion Observer</th>
<th>Galadriel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>Digital camera1</td>
<td>0.93</td>
<td>0.92</td>
</tr>
<tr>
<td>Digital camera2</td>
<td>0.96</td>
<td>0.96</td>
</tr>
<tr>
<td>Cellular phone1</td>
<td>0.93</td>
<td>0.9</td>
</tr>
<tr>
<td>MP3 player</td>
<td>0.87</td>
<td>0.86</td>
</tr>
<tr>
<td>DVD player</td>
<td>0.89</td>
<td>0.88</td>
</tr>
<tr>
<td>Cellular phone2</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>Router</td>
<td>0.84</td>
<td>0.83</td>
</tr>
<tr>
<td>Antivirus software</td>
<td>0.9</td>
<td>0.88</td>
</tr>
<tr>
<td>Total</td>
<td>0.91</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Table 5.7: Overall f-score of OO and Galadriel models.

<table>
<thead>
<tr>
<th>Reviews</th>
<th>Opinion Observer</th>
<th>Galadriel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OO no con-dep no equa</td>
<td>All sent4a sent3</td>
</tr>
<tr>
<td>Digital camera1</td>
<td>0.93 0.91 0.89</td>
<td>0.93 0.91 0.91</td>
</tr>
<tr>
<td>Digital camera2</td>
<td>0.96 0.95 0.95</td>
<td>0.96 0.94 0.94</td>
</tr>
<tr>
<td>Cellular phone1</td>
<td>0.91 0.86 0.88</td>
<td>0.92 0.86 0.88</td>
</tr>
<tr>
<td>MP3 player</td>
<td>0.87 0.83 0.84</td>
<td>0.85 0.83 0.84</td>
</tr>
<tr>
<td>DVD player</td>
<td>0.89 0.87 0.87</td>
<td>0.88 0.86 0.88</td>
</tr>
<tr>
<td>Cellular phone2</td>
<td>0.95 0.92 0.93</td>
<td>0.94 0.9 0.94</td>
</tr>
<tr>
<td>Router</td>
<td>0.83 0.81 0.85</td>
<td>0.84 0.82 0.85</td>
</tr>
<tr>
<td>Antivirus software</td>
<td>0.88 0.81 0.85</td>
<td>0.90 0.8 0.86</td>
</tr>
<tr>
<td>Total</td>
<td>0.90 0.87 0.87</td>
<td>0.90 0.87 0.89</td>
</tr>
</tbody>
</table>
5.5 Discussion

The lexicon-based SO-CAL method produces an overall semantic orientation (polarity and strength) for a document. However, this approach cannot analyse sentiment towards a given product aspect. Moreover, SO-CAL can only handle the words which have already been defined in its dictionaries. It does not have a mechanism for dealing with other words, such as context-dependent sentiment words. The mechanism of SO-CAL involves simply adding the SO value of each word present in a document. Outputs of SO-CAL give numerical values (sum of the SO values of the words). Thus SO values represent not only polarity but also strength. According to this method, a document with a large SO value is stronger than a document with a small SO value. However, this is not always true.

The Opinion Observer (OO) method only produces the polarity of reviews, showing whether the reviews are positive or negative. This method does not concern itself with strength or intensity of sentiment, and it only analyses the sentiment of a review on given product aspects. This approach also focuses on non-sentiment words, which express opinion depending on context. However, how to find the product feature to which the context-dependent opinion words refer, if the word is a feature indicator, is not explained. For instance, ‘The camera is small’. This statement is positive or negative, depending on the polarity (semantic orientation) of small, which can be found from external data. The word small is an indicator of size, but is size a aspect of the camera as a whole? Many components of the camera also have size – the screen, the battery, the memory, etc. The real problem here is that a flat notion of aspects/features doesn’t really work very well, and to attach ‘size’ to as a sub-feature is not explained. Furthermore, OO does not have a mechanism to handle irrealis blocking, which is used by SO-CAL.

In addition, conditional and comparative statements are commonly used in customer reviews, and can be very relevant to sentiment analysis. Both SO-CAL and OO do not have any particular mechanism to deal with conditional and comparison sentences. The SO-CAL feature, irrealis blocking, handles the if statement, but it assumes that these statements are applied in non-factual contexts. However, this is not always true. For example, in the statement ‘if your phone is not good, buy this great Samsung’, the author expresses a sentiment on phone and Samsung. SO-CAL would argue that ‘if your phone is not good’ does not necessarily mean your phone is not good. But it is true that the Samsung is probably great, regardless of the conditional, and SO-CAL cannot capture that. On the other hand, OO would detect that not good refers to ‘phone’. So OO would identify a negative sentiment towards the phone. This issue is handled by Galadriel in a much better
way. In the *Galadriel* lexicon, a list of punctuation is grouped as boundary items (the BOUNDARY node), along with contain words and notations that break sentences into clauses. So, *Galadriel*’s models apply the rules to each lexical entry in clauses, rather than considering the whole document or sentence. This modelling technique breaks the sentence ‘if your phone is not good, buy this great Samsung’ into two clauses. Therefore, in *Galadriel* the first clause is neutral while the second clause show as positive.

[Ding et al. (2008)]’s original OO system extracted the SO value of opinion words in a sentence using the sentiment lexicon dictionary, and used the following equation to aggregate the score values for a particular feature/aspect:

$$\text{Score towards the feature/aspect } f = \sum \frac{w_i \text{SO}}{\text{dis}(w_i, f)}$$

Using the above equation, the SO value of a sentence in regards to each feature can be calculated. Hence, the targeted feature’s score entirely depends on the distance between any opinion words and the targeted feature. However, this is not necessarily always true. For example, if a sentence holds more than one targeted feature, and a mixture of positive and negative opinion words, then it is possible that the equation will go wrong. Consider the following sentence:

‘The only disappointment so far has been battery life, but it has awesome features’

In the above sentence, *disappointment* (a negative opinion word) refers to the targeted aspect battery, and awesome refers to the targeted aspect *features*. However, the distances between *battery* and both negative (*disappointment*) and positive (*awesome*) words are the same. So, according to the equation, the score for the targeted feature, battery, will be 0. Therefore, the proposed equation is not always valid. This problem can be easily overcome by considering the sentences as clauses. As I describe before *Galadriel*’s BOUNDARY items break the sentences into clauses, and *Galadriel* ’s model refers the opinion words to the targeted aspects within the clause.

### 5.6 Tuning and Evaluation

In the previous section, I explained the modelling of *Galadriel* using OO and SO-CAL features in detail, and I discussed the final output of *Galadriel* in chapter 4. This section discusses how that output can be evaluated against the gold standard methods. Unlike machine learning sentiment analysis systems, *Galadriel* does not
produce sentiment classes such as positive, negative or neutral. Similar to most traditional lexicon-based systems, *Galadriel* calculates semantic orientation scores of the lexical items presented in a given document, and, finally, it produces a total score for the document. These systems do a regression task rather than a classification task and sentiment analysis is an ordered classification task. In this way, *Galadriel* returns numerical sentiment scores. Evaluating such systems against data that uses fixed (ordered) classes is difficult. In order to overcome this problem, the numeric data type of the outputs has to be adjusted to the ordered classes. In section 5.6.1 I introduce a tuning method as a pre-evaluation process, which uses a novel calibration technique which shows how to set class thresholds to optimise performance, by using a precision vs recall curve. In section 6.8 of chapter 6, I also identify a parametric feature of the *Galadriel* system, which can control the final *Galadriel* score. Moreover, I describe how I tested the sensitivity and stability of the parametric feature in chapter 6.

### 5.6.1 Pre-Evaluation

Most supervised machine learning methods for sentiment analysis produce categorical outputs such as positive, negative and neutral, with no assumptions about the relationship between classes; they simply map texts into classes by associating text features with class labels. But other multi-class systems use rated or scaled methods so that their categorical outputs are implicitly ordered in a natural ‘sentiment order’ based on sentiment polarity and/or magnitude/intensity, as in the following examples:

- Positive > Neutral > Negative
- Strong-Positive > Positive > Weak-Positive > Neutral
- Weak-Negative > Negative > Strong-Negative
- 3 stars > 2 stars > 1 star

In addition, some sentiment analysis applications are based more explicitly on sentiment scores, rather than sentiment classes. They produce numerical values with positive and negative signs as the output for a given text, such as $+0.987$, $-0.786 \ldots$ or $+187$, $-243 \ldots$ etc. Such methods typically use the sign to indicate the polarity of the given text and numerical values to define the sentiment strength (generally over a system-dependent range), with a sentiment value of 0 indicating a neutral text. A simple mapping from such scores to a 3-class sentiment model uses the sign (+, 0, -) to identify sentiment classes (positive, neutral, negative). However, there is no correspondingly simple way to use the magnitude to extend this to more classes.
(such as strong positive, weak positive, positive, etc.), and no clear justification for the implicit claim that neutral is a single point (0). This section introduces a method to address these concerns, by calibrating the mapping from a numerical score to a semantic class in a way that optimises the system’s performance as a multi-class classifier.

To transform a numeric scale to an ordinal (categorical) scale, boundaries (upper and lower) for each sentiment class need to be identified from the given numeric scale. These boundary values are ‘cut-off values’ for the sentiment classes and are the parameters for a multi-class sentiment classification system based on the numerical scores. This section proposes new techniques to assign cut-off values for each class using a learning-based evaluation technique. This transformation allowed us to both optimise and evaluate a system that gives numeric outputs against a gold standard dataset that contains fixed categorical outputs.

I used evaluation performance measures (precision and recall) on a training subset of the dataset to adjust the parameters to produce an optimal result, by using precision vs recall (PR) curve visualisation. The parameters are optimised to give the best performance on the training set and then evaluated using test set. In addition I can determine how far misclassified texts deviate from actual classes in multi-class ordered classification tasks, by computing macro-averaged mean absolute error which is the popular approach for ordinal classification [Nakov et al. 2016; Baccianella et al. 2009; Gaudette and Japkowicz 2009].

The following demonstrates technique for tuning the parameters using Galadriel. As discussed above, the final output of Galadriel for a text is a signed real number which reflects sentiments expressed by lexical items in quite a complex way, making the interpretation of scores as classes challenging. The calibration method achieves this mapping in an optimal way. Previously, I discussed previous evaluation processes and some general methods involved in sentiment classification in chapter 2. I also produce use of the PR curve for evaluation in section 5.6.1.1. In section 5.6.2, I present my novel techniques for tuning the parameters. In section 5.6.3, I present my experiments with the Galadriel system, and the results of optimising cut-off values for sentiment classes. Section 5.6.4 compares the evaluation results using the cut-off values which are computed in the previous section with evaluation without calibration.

5.6.1.1 The Precision vs. Recall Curve

The use of graphical representations to visualise classifier performance is well-established. The Receiver Operation Characteristic (ROC) curve, originally used in signal detec-
tion theory \cite{Egan1975}, has also been adopted to visualise classifier performances in text classification. The ROC is created by plotting true positive rates (TPR) against false positive rates (FPR) at various thresholds, and the area under the curve has been used as a measure of accuracy in evaluation methods. More recently, researchers have used the precision-recall (PR) curve, which plots precision against the true positive rate, and taken the area under this curve as a measure of performance \cite{West2014, Manning1999, Raghavan1989}. Both curves can be used to visualise classifier performance; however, PR curves produce a more informative visualisation, particularly for highly imbalanced data sets \cite{Davis2006}. A PR curve is more useful for problems where one class is considered to be more important than other classes. On the other hand, there are issues with PR curves too; for example unlike in ROC space it is complicated to interpolate two points in PR space. Furthermore, the area under a PR curve produces the arithmetic mean, whereas the harmonic mean of precision and recall is commonly used to calculate f-score. However, these issues do not affect this work as in the calibration method I only use visualisation of the PR curve to set values for boundaries of sentiment classes.

5.6.2 A Calibration Method for Cut-off Values of Sentiment Classes

In this section, I introduce a calibration method for setting sentiment class cut-off values from numerical sentiment scores using learning-based techniques. I use a training data set to assign boundaries of sentiment classes, where the classes have a natural ‘sentiment order’. This method is inspired by the cross-validation method. I calculate upper and lower boundary values of each sentiment class at a time in sentiment order. For instance, in a three-class classification, I first calculate boundary values for negative (1st class), then neutral (2nd class) and then positive (3rd class). I then determine the optimal cut-off value between these two boundaries to delimit the classes.

To compute the cut-off value, first, I reduce the problem of multi-classes and convert it into the standard binary class problem. That is, I consider the $n^{th}$ order class and the $(n+1)^{th}$ order class to compute the cut-off values between those two classes. I select documents belonging to the $n^{th}$ and $(n+1)^{th}$ classes from the training dataset and run Galadriel over these two sets. As a result, I get a set of numerical scores, one for each document in each class. I consider the maximum Galadriel score for the

\footnote{Such issues can be mitigated by plotting a precision-recall-gain curve \cite{Flach2015} and considering its associated area. However, this is beyond the scope of this work.}
the \( n^{th} \) class, \( Max_n \), and the minimum \( Galadriel \) score for the \((n + 1)^{th}\) class, \( Min_{n+1} \). Then, the cut-off value \((C_{\text{neg/neu}})\) for those two classes can be defined as:

\[
\begin{align*}
Max_n &= C = Min_{n+1} \quad \text{if} \quad Max_n = Min_{n+1} \\
Max_n &\leq C \leq Min_{n+1} \quad \text{if} \quad Max_n \leq Min_{n+1} \\
Max_n &\geq C \geq Min_{n+1} \quad \text{if} \quad Max_n \geq Min_{n+1}
\end{align*}
\]

(5.2)

So the cut-off value, \( C_{n/n+1} \), for those two classes should lie between these two scores.

I plot different PR curves for candidate cut-off values between these scores to determine the cut-off value which gives optimum performance. In order to plot the PR curve, I compute various precision and recall values for each test cut-off value.

\[
\text{Precision(P)} = \frac{tp}{tp + fp}
\]

\[
\text{Recall (R)} = \frac{tp}{tp + fn}
\]

By changing the number of actual document \((tp + fn)\) of a class, the recall value can be changed.

For a given candidate cut-off value, the PR curve plots the \( Galadriel \) system’s ability to classify using that cut-off as the class boundary, for different mixtures of the two classes. The data set is divided into \( k \) subsets (folds) with an equal number \((d)\) of documents. I assume the data set is normally distributed. Each subset contains \( n^{th} \) class documents and \((n + 1)^{th}\) class documents in different proportions. For example, the 1\(^{st}\) subset contains \( m_1 \) number of \( n^{th} \) class documents and \((d - m_1)\) number of \((n + 1)^{th}\) class documents, the 2\(^{nd}\) subset contains \( m_2 \) number of \( n^{th} \) class documents and \((d - m_2)\) number of \((n + 1)^{th}\) class documents, and the \( k^{th}\) subset contains \( m_k \) number of \( n^{th} \) class documents and \((d - m_k)\) number of \((n + 1)^{th}\) class documents (see figure 5.22a). Each fold represents a different distribution of sentiment scores for the two classes (see figure 5.22b) and hence a different precision and recall score for each class for the given cut-off. I then calculate the macro-average precision and recall across the two classes; the PR curve plots these different precision/recall values for a single cut-off value across all the folds.

The best cut-off value produces high and almost equal values of precision and recall. Therefore, the PR curve of the best cut-off value lies to the top right hand corner.

\footnote{Note that the classes’ score ranges may overlap: \( Max_n \) may be greater than \( Min_{n+1} \).}
(a) Folds containing mixtures of 2 document classes

(b) Example histograms and cut-off points for different balances between $n^{th}$ and $(n+1)^{th}$ classes in different folds

Figure 5.22: $k$-fold class mixtures, to produce PR curves for each cut-off candidate
of the graph as well as close to the diagonal line \((p = r)\). I originally hoped that I could choose the best PR curve by visual inspection, but in practice, while this is sufficient to rule out many candidates, the final choice was also supported by additionally plotting average recall and precision for each PR curve.

Once the best cut-off value, \(C_{n/n+1}\), has been established, the process is repeated for the other class boundaries \((C_{n+1/n+2} \text{ etc.})\). These cut-off values can then be used to map the numerical scores to classes in an optimal way. For example, in the three class \textit{negative}, \textit{neutral}, \textit{positive} case, with classes 1, 2 and 3, I use \(C_{1/2}\) as the boundary between \textit{negative} and \textit{neutral}, and \(C_{2/3}\) as the boundary between \textit{neutral} and \textit{positive}, and classify as follows:

\[
S_i = \begin{cases} 
  \text{positive}, & \text{if } \text{Tot}_i > C_{2/3} \\
  \text{neutral}, & \text{if } C_{1/2} < \text{Tot}_i < C_{2/3} \\
  \text{negative}, & \text{if } \text{Tot}_i < C_{1/2} 
\end{cases} \tag{5.3}
\]

where \(S_i\) is the sentiment class of document \(i\) and \(\text{Tot}_i\) is the total sentiment score of the document \(i\).

\section*{5.6.3 Experiments and Results}

To test the above method, I experimented with the \textit{Galadriel} sentiment analysis system on a scaled dataset\footnote{http://www.cs.cornell.edu/people/pabo/movie-review-data/} used by Pang and Lee\,(2005). The dataset is a collection of movie reviews labelled with values of 0, 1 and 2. When analysed by the \textit{Galadriel} system, the documents in this dataset return scores ranging between \(-10\) and \(+25\).

The purpose of this experiment was to show that by assigning optimal cut-off values for \textit{Galadriel} scores according to this scaled dataset, the system’s output can be mapped into this three-class system in a way which maximises its performance as a sentiment classifier.

I selected 300 documents of approximately equal length from the dataset (100 documents for each scale value in an approximately normal distribution). First I divided the dataset into two parts, one for training and another for testing. I used 240 documents (80 documents from each scale) as my training set. First, I computed boundaries for the \textit{scale-0} class, then for the \textit{scale-1} class and finally for the \textit{scale-2} class. Since \textit{scale-0} is the lowest class it is not necessary to compute the lower boundary for \textit{scale-0}. To determine the upper boundary of the \textit{Galadriel} score for \textit{scale-0}, the cut-off value of the \textit{Galadriel} score between \textit{scale-0} and \textit{scale-1} needed to be computed. For this, I used my \textit{scale-0} and \textit{scale-1} training documents (160
documents). I found that the maximum normalised *Galadriel* score for *scale-0* documents was +0.17 and minimum *Galadriel* score for *scale-1* documents was −1.41 (rounded up to two decimals). Therefore, I set up candidate cut-off values ($C_i$) between −1.45 and +0.2 in an equal interval of 0.05, i.e., −1.45, −1.40, −1.35, −1.30, −1.25, −1.20, −1.15, −1.10, −1.05, −1.00, −0.05, 0.00, +0.5, +0.1, +0.15, +0.2.

Then, for each candidate cut-off value, I calculated precision and recall values were calculated for 5 sub-training data sets, each subset containing a mixture of 32 *scale-0* and *scale-1* documents. For each cut-off value ($C_i$) precision and recall values were calculated for the *scale-0* and *scale-1* classes. Then the precision and recall values were summarised by taking the macro-average of both classes’ values. Finally, I had 5 pairs of precision and recall values for each of my 28 candidate cut-off values. Figure 5.23 shows the resulting 28 different PR curves. The ideal cut-off value will have a PR curve as close to the diagonal, and as far towards the top right corner, as possible. As can be seen in figure 5.23, although the general trend is for all the curves to be in the top right half of the graph, many of them deviate significantly from the diagonal line. I focused on the six curves closest to the diagonal (by visual inspection), shown in figure 5.24, for further analysis.

The 6 candidate cut-off values remaining after this step are −0.75, −0.70, −0.65, −0.60, −0.55 and −0.50. The PR curves of those values lie closest to the diagonal and largely in the upper-right corner. Thus I concluded that one of those 6 test values is the optimal cut-off value $C_{0/1}$ for the *scale-0* and *scale-1* classes.
Figure 5.24: Most appropriate PR curves

Figure 5.25: Average of Precision and Recall values
Cut-Off Values | Recall | Precision | F-Score
---|---|---|---
-0.75 | 0.7480 | 0.8020 | 0.7741
-0.70 | 0.7712 | 0.8046 | 0.7875
-0.65 | 0.8131 | 0.7972 | 0.8050
-0.60 | 0.8164 | 0.7922 | 0.8042
-0.55 | 0.8250 | 0.7851 | 0.8046
-0.50 | 0.8250 | 0.7608 | 0.7916

Table 5.8: Average precision, recall and f-score measures for candidate cut-off values

<table>
<thead>
<tr>
<th>Folders</th>
<th>Cut-Off Values</th>
<th>Performance Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$C_{0/1}$</td>
<td>$C_{0/1}$</td>
</tr>
<tr>
<td>Iteration-1</td>
<td>-0.65</td>
<td>1.05</td>
</tr>
<tr>
<td>Iteration-2</td>
<td>-0.55</td>
<td>1.35</td>
</tr>
<tr>
<td>Iteration-3</td>
<td>-0.70</td>
<td>1</td>
</tr>
<tr>
<td>Iteration-4</td>
<td>-0.8</td>
<td>0.75</td>
</tr>
<tr>
<td>Iteration-5</td>
<td>-0.75</td>
<td>1.2</td>
</tr>
</tbody>
</table>

Table 5.9: Cross-validation process for calibration system

Looking more closely, I can see that the PR curves for $-0.65$, $-0.60$, $-0.55$ and $-0.5$ lie noticeably closer to the top right-hand corner compared to the PR curves for $-0.75$ and $-0.70$. I therefore discard these two, but the remaining curves track each other very closely - too closely for visual discrimination. I therefore calculated the (macro-)average precision and recall values of each cut-off value and plotted these in a scatter plot (figure 5.25). From this plot, it can be concluded that the best cut-off value for the $scale-0$ and $scale-1$ classes is $-0.65$.

To validate this cut-off value, I also compared f-scores for the candidate cut-off values from these macro-averaged recall and precision values. I only considered the candidate values used in figure 5.24, as the remaining cut-off values had already been rejected. Table 5.8 also shows these numbers for the different candidate cut-off values. The f-score of the cut-off value $-0.65$ has the maximum value.

Similarly, the cut-off value $C_{1/2}$ for the $scale-1$ and $scale-2$ classes was computed with an optimal value of $+1.05$.

To get the optimal performance, I ran Galadriel over 60 documents from the test dataset with the computed two cut-off values and calculated the f-score and the MAE values. This process were repeated five times by choosing various sets of training and testing data. I finally obtained five different pairs of cut-off values and relevant f-score and MAE values for the test dataset as shown in table 5.9. Since iteration 1 produces better f-score and MAE values compare to other iterations, I assumed cutoff values $-0.65$ ($C_{0/1}$) and $+1.05$ ($C_{1/2}$) lead to better performance.
<table>
<thead>
<tr>
<th>Galadriel Scores of Documents</th>
<th>Scaled Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>−0.65 &gt; Gal_i</td>
<td>0 1 2</td>
</tr>
<tr>
<td>−0.65 &lt; Gal_i &lt; +1.05</td>
<td>3 16 2</td>
</tr>
<tr>
<td>+1.05 &lt; Gal_i</td>
<td>2 3 18</td>
</tr>
</tbody>
</table>

Table 5.10: Confusion matrix for the classification

<table>
<thead>
<tr>
<th>Document Scales</th>
<th>Calibrated System</th>
<th>Uncalibrated System</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>scale-0</td>
<td>0.9375</td>
<td>0.7500</td>
</tr>
<tr>
<td>scale-1</td>
<td>0.7619</td>
<td>0.8000</td>
</tr>
<tr>
<td>scale-2</td>
<td>0.7826</td>
<td>0.9000</td>
</tr>
<tr>
<td>Macro-average</td>
<td>0.8273</td>
<td>0.8167</td>
</tr>
</tbody>
</table>

Table 5.11: Comparing performance measures calculated by the calibrated and uncalibrated versions of Galadriel.

### 5.6.4 Evaluation of the Calibrated System

In order to demonstrate the effect of the calibration process, I evaluated the calibrated Galadriel system against Pang and Lee (2005)’s dataset and compared this with the evaluation of the uncalibrated version. For this evaluation, 50 random unseen test documents from the dataset were selected and analysed them using Galadriel, giving numerical scores for each document as its output. The output scores were classified according to Galadriel cut-off values −0.65 (C₀₁) and +1.05 (C₁₂). Table 5.10 shows the resulting confusion matrix. It is interesting to note that this optimum score range for the neutral class is quite small in comparison to the total score range of the system (1.70 out of 30), and also is not balanced around zero.

Table 5.11 shows precision, recall and f-score results for each class and overall macro-average results, for both the calibrated system and the uncalibrated system, which maps sentiment scores simply on the basis of their sign (negative, zero or positive). In the uncalibrated system, the Galadriel score of the scale-1 class documents was expected produce 0. However, very few documents produced the total score of 0 due to the different levels of Galadriel models and the calculations. This impacts the overall performance (macro average of f-score) of Galadriel’s uncalibrated system, which showed very poor results. Then the effect of calibrating is to increase the macro-averaged f-score from 0.48 to 0.82. Moreover, the calibrated system gives overall macro-averaged mean absolute error (MAE) of 0.2167 whereas the uncalibrated system shows 0.5166.
5.6.5 Discussion

This section presented a novel calibration method for transforming numerical sentiment scores into fixed, ordered classes. This method uses corpus-based evaluation techniques, widely used in supervised machine learning approaches, calibrating a system using gold standard labelled data. The effect is to optimise a continuous sentiment analysis system for the discrete classification model represented by the gold standard data. The calibrated system can then be evaluated and compared with other systems by using additional unseen gold standard data for the same model, or applied to new data assumed to follow the same model, with the confidence provided by the evaluation results. The availability of a general calibration method also means that the same system can be calibrated independently for different classification tasks as required.

I also presented a comparison between the performance of a calibrated system and the corresponding uncalibrated system, where sentiment scores are mapped into classes based solely on their sign, and showed that calibration can provide a substantial increase in performance. Although the uncalibrated system might be considered a poor baseline for comparison, it is worth bearing in mind that it is a simple model such as this which often guides the assignment of lexical semantic orientation scores. The effectiveness of calibration is a measure of the extent to which the document analysis process as a whole deviates from the simple lexical model, in a way that is difficult to capture by other means, and reveals interesting biases in the way the process maps sentiment onto scores.

5.7 Evaluation of Modelling SO-CAL and OO in *Galadriel* Using the Calibration Method

Sections 5.1 and 5.3 show how the SO-CAL and OO features were implemented in *Galadriel* in detail, followed by a comparison of evaluations. Then I introduced new evaluation techniques that could improve the evaluation results of sentiment analysis systems. I recalculated the evaluation metrics of the *Galadriel* system, which was previously modelled by SO-CAL and OO features.

Both the SO-CAL and OO systems are binary sentiment analysis systems. They classify the sentiment of a text into positive and negative. Similar to the *Galadriel* system, SO-CAL and OO also calculate an SO value and produce numeric scores as a final output, labelled **positive** if the final score is greater than 0, otherwise **negative**. I wanted to calibrate the *Galadriel* numeric score with the original (Gold
Table 5.12: Performance comparison of Galadriel with calibration method on SO-CAL features

<table>
<thead>
<tr>
<th>Reviews</th>
<th>SO-CAL</th>
<th>Galadriel</th>
<th>Galadriel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pos-F</td>
<td>Neg-F</td>
<td>Uncalibrated Pos-F</td>
</tr>
<tr>
<td>Books</td>
<td>0.69</td>
<td>0.74</td>
<td>0.73</td>
</tr>
<tr>
<td>Cars</td>
<td>0.9</td>
<td>0.89</td>
<td>0.89</td>
</tr>
<tr>
<td>Computers</td>
<td>0.94</td>
<td>0.94</td>
<td>0.95</td>
</tr>
<tr>
<td>Cookware</td>
<td>0.74</td>
<td>0.58</td>
<td>0.8</td>
</tr>
<tr>
<td>Hotels</td>
<td>0.76</td>
<td>0.67</td>
<td>0.79</td>
</tr>
<tr>
<td>Movies</td>
<td>0.84</td>
<td>0.84</td>
<td>0.89</td>
</tr>
<tr>
<td>Music</td>
<td>0.82</td>
<td>0.82</td>
<td>0.85</td>
</tr>
<tr>
<td>Phones</td>
<td>0.81</td>
<td>0.78</td>
<td>0.81</td>
</tr>
<tr>
<td>Total</td>
<td>0.81</td>
<td>0.79</td>
<td>0.84</td>
</tr>
</tbody>
</table>

standard) datasets used in the SO-CAL and OO systems, and get the cut-off values of each model. I then re-classified the sentiment of the documents according to the cut-off values.

I calculated a cut-off value of +0.76 for Galadriel with SO-CAL features. Thus the documents were classified as follows:

**Negative** : if Galadriel score < +0.76

**Positive** : if Galadriel score > +0.76

Similarly, the cut-off value for Galadriel with OO features is +0.17 and classified as follows:

**Negative** : if Galadriel score < +0.17

**Positive** : if Galadriel score > +0.17
<table>
<thead>
<tr>
<th>Reviews</th>
<th>OO</th>
<th>Galadriel Uncalibrated</th>
<th>Galadriel Calibrated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital camera 1</td>
<td>0.93</td>
<td>0.93</td>
<td>0.94</td>
</tr>
<tr>
<td>Digital camera 2</td>
<td>0.96</td>
<td>0.96</td>
<td>0.96</td>
</tr>
<tr>
<td>Cellular phone 1</td>
<td>0.91</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>MP3 player</td>
<td>0.87</td>
<td>0.85</td>
<td>0.88</td>
</tr>
<tr>
<td>DVD player</td>
<td>0.89</td>
<td>0.88</td>
<td>0.88</td>
</tr>
<tr>
<td>Cellular phone 2</td>
<td>0.95</td>
<td>0.94</td>
<td>0.95</td>
</tr>
<tr>
<td>Router</td>
<td>0.83</td>
<td>0.84</td>
<td>0.84</td>
</tr>
<tr>
<td>Antivirus software</td>
<td>0.88</td>
<td>0.90</td>
<td>0.90</td>
</tr>
<tr>
<td>Total</td>
<td>0.90</td>
<td>0.90</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Table 5.13: Performance (f-score) comparison of Galadriel with calibration method on OO features

The final f-scores show that the calibration evaluation method improved the evaluation results.

5.8 Summary of the Chapter

This chapter has discussed two existing lexical-based approaches to sentiment analysis. One approach is document-level analysis, and the other is aspect-level analysis. It has been showed that these methods could be modelled in Galadriel using inheritance-based techniques. These models were evaluated by comparing the existing original methods. From these analyses, an inheritance model of sentiment knowledge of words was identified and extended to a model of sentiment analysis. In this way, the entire sentiment analysis task could be coded as a ‘lexical description’ task.

Additionally, this chapter explained an evaluation method for the Galadriel system. Before the evaluation process, the final Galadriel numeric scores have to be adjusted to the fixed classes, which are used in the gold standard method. Therefore, a calibration method needed to be introduced. Section 5.6.1 proposed a novel calibration technique by using an example-based evaluation technique (precision vs recall curve) to set class thresholds. I also showed how we tested the class thresholds using a movie review dataset. It has been showed that the f-score of the Galadriel system improved to 0.8220 from 0.4838, and averaged mean absolute error (MAE) produces 0.2167 from 0.5166 on a sample dataset, when the calibration method was used.

124
Finally, I updated the modelling of SO-CAL and OO features in *Galadriel* evaluations using the calibration method. I demonstrated that the final *Galadriel* model gives better results which are f-score of 0.86 and 0.83 for positive and negative reviews on SO-CAL features respectively and f-score of 0.91 for the overall dataset for OO features.
Chapter 6

Implementation of An Integrated Model

In chapter 5, I modelled sentiment knowledge using DATR’s inheritance mechanism by modelling SO-CAL and Opinion Observer (OO) in Galadriel. I aimed to adopt the rules and algorithms which were used to model sentiment knowledge in their work. I also identified required improvements by evaluating the effectiveness of the models. In this chapter, an integrated model is developed by extending the inheritance-based model of sentiment knowledge to model four distinguished sentiment analysis tasks: word-level, phrase-level, sentence-level and document-level. Sentiment dictionaries (especially sentiment lexicon) were used for the word-level task. I use corpus-based learning techniques to populate sentiment phrases from example corpus data. For the sentence and document levels, I develop models using appropriate rules and algorithms. This chapter discusses the modelling approach to sentiment analysis in Galadriel using novel techniques.

I focus on developing novel techniques in three main areas, using the inheritance-based structure I developed in chapter 5:

(i) Modelling of the sentiment lexicon in an inheritance-based structure based on sentiment behaviour of lexical items.

(ii) Extend the lexicon with sentiment phrases and irregular sentiment behaviour of lexical items by exploiting corpus-based learning techniques and other available lexicons.

(iii) Development of various sentiment models that are inherited from each other.

This chapter explains how the final system (Galadriel version 1.0) is developed for
sentiment analysis by combining the techniques that I developed for the above areas.

6.1 An Integrated Model

The integrated model is a combination of both SO-CAL and OO with some added techniques. Galadriel has a set of ‘feature:value’ pairs for each lexical item, which define its sentiment behaviour. The final Galadriel system is a collection of models which calculate the sentiment behaviour or semantic orientation of a piece of text according to its grammatical structure. Each model has a lexical agent that has rules and algorithms to handle lexical valence and contextual valence shifters. Moreover, each model is inherited from the other. The basic rules and algorithms of SO-CAL and OO were adopted to develop the integrated model. This section shows the implementation of the document/sentence-level sentiment analysis system. I start with modelling the Galadriel lexicon or ‘base-model’, in which the lexical items are modelled at the word level. Then I discuss the other models, which are named sent1 model, sent2 model, sent3 model, etc., which model lexical items at sentence and document levels.

6.1.1 Galadriel Base Model: Galadriel Lexicon

This section discusses the modelling of sentiment analysis at the word-level by exploiting an inheritance-based lexicon. In other words, I discuss the modelling of the sentiment lexicon in Galadriel by using (Taboada et al., 2011)’s sentiment lexicon and sentiment scores. As explained in chapter 4, a word (or lexical item/lexical entry) is considered to be a lexical agent (or automaton). Each lexical agent has a set of feature:value pairs which define its sentiment behaviour. In addition, we introduced the following new features to the base Galadriel model:
Figure 6.1: The NEUTRAL node of the integrated *Galadriel* lexicon with updated feature values

- **polarity**: This refers to the lexical item’s polarity type. The value takes only +1, −1 and 0 for positive, negative and neutral lexical items, respectively.

- **magnitude**: This feature indicates the sentiment intensity of the lexical item. Following Taboada et al.’s sentiment scores, ranging from +5, −5, I assign the value 5 for very strong positive/negative lexical items (e.g. excellent, worst), and 1 for very weak positive/negative lexical items (e.g. wacky, crooked).

- **context-pol**: The context-pol feature of a lexical item is used to identify its polarity within the sentence (or its contextual sentiment). Similar to the polarity feature, context-pol also takes values +1, −1 or 0.

- **senti-score**: The senti-score of a lexical item is a combination of its magnitude and polarity. This is the most important feature for the lexical item on any sentiment analysis task and it decides the final sentiment of the lexical item. The value is not populated from the sentiment dictionary but it is calculated using the following equation:

\[
\text{senti-score} = \text{magnitude} \times \text{polarity}
\]

For the integrated model I replace the score feature with polarity, magnitude and senti-score, in order to make it a fine-grained model. This allows models to
calculate magnitude and polarity separately. Moreover, the new feature `context-pol` value provides the polarity of a word within its context.

For the integrated model, I also made some changes in the *Galadriel* base model by adding extra abstract nodes that are inherited from NEUTRAL, such as CONTEXT and QUEST. Figure 6.1 shows the updated hierarchical lexicon structure with the added default feature values.

Moreover, I added child nodes to the POSITIVE and NEGATIVE nodes, that can group the sentiment magnitude values. This section explains the changes that I made to the previous *Galadriel* version (in chapter 4) in order to model the integrated model.

### 6.1.1.1 POSITIVE and NEGATIVE Nodes

```plaintext
galadriel.sentiment.ROOT:
<> == galadriel.LEXROOT
<base type> == neutral
<base pol> == 0
<base mag> == 0
<base factor> == 1
<base senti-score> == 0
<base contex-pol> == 0
<neg-context> == no
<base block-context> == no
<base ques-context> == no
<base total> == 0

galadriel.sentiment.POSITIVE:
<> == galadriel.sentiment.ROOT::<>
<base type> == positive
<base pol> == +1
<base mag> == 0

.  

galadriel.sentiment.NEGATIVE:
<> == galadriel.sentiment.ROOT::<>
<base type> == negative
<base pol> == -1
<base mag> == 0
```

Figure 6.2: The *Galadriel* code for the POSITIVE and NEGATIVE nodes

Chapter 4 described how the POSITIVE and NEGATIVE nodes inherit from the abstract SENTIMENT node (it’s called ROOT in the code), which is the root node in the higher-level structure. In the integrated model, I further divide the positive and negative words by their sentiment strength (magnitude values). Figure 6.2 shows
the *Galadriel* code for the root *SENTIMENT* node with the child nodes *POSITIVE* and *NEGATIVE* nodes as modelled in *Galadriel*.

I used Taboada et al.’s (2011) dictionaries to assign magnitude value of both positive and negative words. Both positive and negative polarity classes can have subclasses depending on the magnitude values of the words. I created abstract nodes for subclasses that inherit from *POSITIVE* and *NEGATIVE* (see figures 6.3 and 6.4). For example, the feature values of *POSITIVE* inherit to its child nodes *VERY STRONG POSITIVE*, *STRONG POSITIVE*, *POS*, *WEAK POSITIVE* and *VERY WEAK POSITIVE*. In every child node, the value of magnitude is overridden by sentiment strength, as shown in figure 6.3. The type value of these child nodes is only used for the explanation and I do not add this feature in the implementation of modelling. This is because, when a model calls for the lexical item that has the type value *positive*, *Galadriel* would not identify it as a positive word as it is not directly inherited from the *POSITIVE* node. Therefore, I only use the magnitude feature of these nodes.

![Diagram of sentiment hierarchy](image)

Figure 6.3: Positive lexical items in the hierarchy
As mentioned above, I used Taboada et al.’s (2011) sentiment lexicons\(^1\) to group the lexical items into the magnitude nodes depending on their positive/negative strengths. For example, the magnitude value of the subclass VERY STRONG POSITIVE (V.STR POS) is represented as:

\[
\text{magnitude} := 5
\]

Figure 6.5 shows some positive words as modelled in \textit{Galadriel}. For example, consider the lexical item \textit{excellent}; its feature values are inherited from the top level of the hierarchy. The SENTIMENT node is assigned with default feature values in the base model. Then the feature values are passed down to the lower level, while some feature values (shown in bold letters in the figure) are overridden in the different nodes, as shown in figure 6.5.

Let us consider the lexical item \textit{excellent}, which is a lexical agent and inherits its feature:value pairs from the node V.STR POS, as follows:

\(^1\)The SO-CAL dictionary contains a list of lexical items (adjectives, adverbs, nouns and verbs) with their SO (semantic orientation) values (between -5 and +5)
Figure 6.5: The feature values with small font size are passed down through the hierarchy and the feature values with large font size are overridden in the Galadriel base model.

Figure 6.6 shows how some positive words are modelled in the base model using Galadriel code in the lower level of the hierarchy. The magnitude value of each lexical item is overridden by its own magnitude values. Similarly, other words are
modelled in the base model with appropriate feature values.

6.1.1.2 CONTEXT Node

As discussed above, some neutral words show sentiment polarity depending on their context. I call such words context-dependent words (context words). For example, long and short are neutral words. However, short shows negativity in the sentence ‘Battery life is short on the phone’, and positivity in the sentence ‘The machine completes the job in a short period of time’. I have collected such neutral descriptive adjectives\(^2\) and grouped them together to create a subclass of neutral words. Thus context words are represented by a node called CONTEXT which is inherited from the node NEUTRAL (figure 6.7). The feature values of NEUTRAL inherit to the node CONTEXT, but type and magnitude values are overridden by context and 1, respectively. In section 6.1.3 I explain how Galadriel handles these context words in detail.

Furthermore, I also tried to assign individual magnitude values for context words depending on their strength or size. For instance, consider the related context words huge, large and big. Although those three words have a similar definition,
Figure 6.7: Context words show sentiment depending on their context

Figure 6.8: Context words can have different magnitude values
their weights of meaning are different. That is to say, huge is very, very big, large is very big. Therefore, I assigned a different value for each word depending on its weight of meaning. For example:

\[
huge_{\text{mag}} = 3 \\
large_{\text{mag}} = 2 \\
big_{\text{mag}} = 1
\]

This can differentiate the sentiment meaning of those three words. Similarly, I collected other synonyms of context words from web dictionaries\(^3\) and assigned their magnitude values. This was structured in the Galadriel inheritance structure as shown in figure 6.8. The feature values of every context-dependent lexical-agent(words) are inherited from CONTEXT and override magnitude value by their own value.

### 6.1.1.3 BOUNDARY Nodes

In order to handle negation and modality, the sentences and clauses which are in the scope of negators and irrealis markers have to be identified. I already have a list of punctuation marks\(^4\) modelled under the BOUNDARY node. For the integrated model, I also collected a list of words that can break sentences into clauses. Generally, a document or paragraph can be separated into sentences using punctuation marks. Simple sentences contain only one independent clause; however, complex sentences contain more than one, the main clause and one or more dependent clauses (Brinton, 2000). I added some words such as and, although, however and but, which can also separate clauses. These words were grouped and modelled under the BOUNDARY node.

However, some boundary words exhibit different behaviour to others. They are not always used to break sentences into clauses; they can also be used to connect two words or phrases. In such situations, those words should not be considered boundary words, and they are treated as neutral words. For example, and and ’,’ do not behave like BOUNDARY words when they are present in between the same part-of-speech tagger. For instance, and is not a BOUNDARY word in the following sentence:

‘The game is suitable both for children and adults.’

\(^3\)https://wordnet.princeton.edu/ and https://www.thesaurus.com/
\(^4\)https://www.grammarbook.com/
In the above sentence, *and* is present in between two nouns (*children* and *adults*) and it does not break the sentence, like other BOUNDARY words. So the *type* values of *and* and *‘,’* take either *neutral* or *boundary*, depending on their neighbouring lexical items. This can be technically defined using DATR/ELF in the *Galadriel* dictionary (*Galadriel* base model). The following algorithm were used to model such words (This was coded in *Galadriel* as shown in figure 6.9): for *and* and *‘,’*:

```plaintext
if POS tag of prev word = POS tag of next word then
    type = neutral
else
    type = boundary
end if
```

Figure 6.9: The *Galadriel* code: the lexical items neighbouring *and* and *‘,’* are checked for if they contain same part-of-speech tag.

### 6.1.1.4 QUESTION Node

An interrogative sentence is a statement that asks a question, and always ends in a question mark (*?*). Such sentences do not show any opinion or sentiment. Therefore, any polar (positive or negative) words present in an interrogative sentence can be ignored. SO-CAL dealt with interrogative sentences using its irrealis feature. However, *Galadriel’s* integrated model uses a slightly different method, by exploiting interrogative words.

In order to identify interrogative sentences, I collected a list of interrogative lexical items (for example *what*, *when*, *how*). I grouped them together and structured them under an abstract node called QUESTION, which inherits from NEUTRAL. All the feature values (except *ques-context*) of QUESTION are inherited from NEUTRAL and the feature *ques-context* is overridden by *yes*. However, interrogative words do not necessarily always appear in a question statement. For example, consider the
below example:

‘The camera is very small, which I don’t like.’

In the above statement, which does not indicate a question. This can be easily identified by checking whether there is a question mark at the end of the sentence. Also, questions mostly start with an interrogative lexical item. I added rules to the Galadriel model to detect if a specific interrogative lexical item indicates a question or not. Then the lexical agent can be overridden by the value yes. Some questions start with verbs such as is, are, etc. These types of questions are identified by checking if these verbs are present at the beginning of the sentence.

6.1.2 Extending the Galadriel Lexicon

The Galadriel lexicon was mainly created using the SO-CAL dictionary. However, I extended the lexicon by adding more words or phrases that have an impact on the sentiment of a document/sentence. I exploited two methods to access external lexicons, which are discussed in this section.
6.1.2.1 Exploiting Corpus-Based Learning Techniques

Like any other typical lexicon-based method, Galadriel relies on its sentiment lexicons, which were modelled in an inheritance-based structure based on the behaviour of the lexical items, as discussed in section 6.1.1. However, there are some lexical items which show different or irregular sentiment behaviour based on context or domain, which may not available in current Galadriel lexicons. Moreover, for some sentiment analysis tasks, such as aspect-level tasks, I need various lexicons to run Galadriel. For example, aspect terms and indicators should be plugged into the Galadriel lexicon in order to run Galadriel’s aspect-based model, as it is not automatically detecting aspects terms. Therefore, I needed to extend the Galadriel lexicon. However, populating lexicons is a challenging task. Therefore, I aimed to use a corpus-based approach to populate various type of lexicons. Corpus-based learning is an approach that utilises an underlying corpus as an index of language data. I use Sketch Engine\(^5\) which is a corpus analysis tool, to populate lexicons (made up of single words and phrases) from example reviews (used as training dataset), and thus extend the Galadriel lexicon.

\(^5\)https://old.sketchengine.co.uk/open/
**Sketch Engine:** Analytical software and a corpus manager that takes as its input a multi-language corpus. It generates word sketches based on its language grammar patterns, which are useful for people who study language behaviour. Sketch Engine was developed by research scientist Dr Adam Kilgarriff[^1]. Sketch Engine became a commercial software product of Lexical Computing Limited in 2003. I used Sketch Engine’s listing word feature to list single words and phrases present in a corpus.

Training datasets (or example reviews) were used to populate required list of words or phrases. Sketch Engine lists the words/phrases present in the dataset with their frequency, as shown in figure 6.12. I used positive and negative reviews and checked if there was any irregular behaviour or special words present.

**Irregular Sentiment Lexicon:** I discussed neutral words that behave as sentiment words depending on their context (context-dependent sentiment words) in 6.1.1.2. Similarly, some sentiment lexicons behave as neutral, depending on context. Consider the following example:

> ‘I am going to leave the company for good.’

In the above example, the sentiment lexical item good behaves as a neutral word, because of its neighbouring lexical item (*for*). This makes the phrase for good neutral. I populated such sentiment lexical items that show different sentiments.

[^1]: [http://kilgarriff.co.uk/](http://kilgarriff.co.uk/)
when they appear with a specific word or set of words from training datasets. Then I modelled them as phrases and assigned appropriate sentiment labels. I used various examples/corpora as training datasets. Then the phrases were modelled and added under the relevant polarity node in Galadriel’s hierarchical inheritance structure. For example, in figure 6.11, the phrase for good is considered as one word.

**Domain-Specific Lexicon:** Some lexical sentiment items are present as part of a noun phrase that behaves as a neutral sentiment. For example:

‘Bad Boys stars Will Smith and Martin Lawrence, who have enjoyable chemistry.’

In the above example, the phrase Bad Boys indicates a movie name. However, this behaviour is only limited to a specific domain and a specific movie review. I use sample reviews from the same domain and use Sketch Engine to populate such phrases and add them to the Galadriel lexicon.

Similarly, for document-level sentiment analysis, some lexicons have different sentiment scores/semantic orientations in different domains. For example, the adjective unpredictable has a positive score in the movie domain, as in the phrase ‘unpredictable plot’, which exhibits a positive sentiment. Such phrases were labelled with a sentiment score and extracted from a training dataset from a specific domain. Then the phrases can be modelled in Galadriel hierarchical inheritance lexicon structure. Moreover, similar to Wilson et al. (2005), intensifier and negation features also can be modelled as phrases in Galadriel. However, I did not use phrase models to model intensifiers and negation words in Galadriel.

```
galadriel.word.N-bad: <> == galadriel.sentiment.NEGATIVE <base type> == IFEQ :< <here base next base word .> boys THEN neutral ELSE negative .> <base mag> == 3 .
galadriel.word.N-unpredictable: <> == galadriel.sentiment.NEGATIVE <base type> == IFEQ :< <here base next base word .> plot THEN positive ELSE negative .> <base mag> == 3 .
```

Figure 6.13: Phrases are modelled in the base model
Modelling Sentiment Phrases: As discussed above, some polar lexical items express a neutral sentiment when they appear next to a specific word/words. Appropriate values override the values of such lexical items in the base model. Similarly, domain-specific phrases are also modelled in the base model. Consider the same examples as used above:

for good: a phrase used to express forever
Bad Boys: a movie name
unpredictable steering: a phrase in the automotive review domain
unpredictable plot: a phrase in the movie review domain

I explained the sentiment behaviour of the above phrases earlier in the chapter. Figure 6.13 shows the Galadriel code for the above phrases in the base model.

6.1.2.2 Exploiting Existing Lexicons

Sentiment Idioms: An idiom is a group of words which gives a specific meaning. However, the meaning of the idiom is different from the meanings of each word on its own. Most idioms are used to show an opinion or sentiment without having a single sentiment word in them. Consider the example:

‘When I saw your message, I was over the moon.’

No sentiment words are present in the above sentence. Nevertheless, the sentence shows a positive sentiment, because the phrase over the moon means happiness, informally. Sentiment idioms can be populated via corpus-based learning techniques. I used (Williams et al., 2015)’s list of sentiment idioms for this experiment. Similar to section 6.1.2.1, each sentiment idiom was considered as a single unit and modelled in Galadriel as a subclass of the appropriate polarity node, as shown in figure 6.11

Modelling Sentiment Idioms: As explained above, sentiment idioms are groups of words that express sentiment. Consider the following examples:

1. over the moon
2. the bee’s knees
3. kiss of death

To model sentiment idioms/phrases, I chose one or two of the words present in the phrase and added certain conditions to it. For example, for phrase 1, I consider the lexical item moon, and apply a special rule for assigning its type value in Galadriel.

[https://users.cs.cf.ac.uk/I.Spasic/idioment/](https://users.cs.cf.ac.uk/I.Spasic/idioment/)
Figure 6.14: Sentiment phrases are modelled in the base model by considering one or two lexical items of the phrase

The following rule is used for this:
For word, moon

if prev word =the then

if prev prev word =over then

type = positive;
else

type = NEUTRAL;
end if
else

type = NEUTRAL;
end if

I chose only lexical items that are not a stop word to model sentiment phrases in Galadriel, in order to save the execution time. Moreover, if a polar word is present in a sentiment phrase, I add special rules to the polar lexical item. For instance, phrase 3 above has a negative lexical item, death, with a magnitude feature value 1. However, the whole phrase kiss of death expresses a negative sentiment with a magnitude value of 4. Figure 6.14 shows how such sentiment phrases are modelled in the base model.

6.1.3 Development of Sentiment Models in Galadriel

I have discussed the implementation of modelling lexical items in the Galadriel lexicon, which is called the Galadriel base model of the integrated model. The base
model calculates the sentiment score of lexical items at the word level. Similar to the Galadriel systems that I modelled in chapter 5, the integrated Galadriel system has six different models (sent1 to sent6). Each model has different rules and algorithms to calculate the sentiment score of lexical items at the sentence/document level. This section discusses all six models of Galadriel’s integrated system.

### 6.1.3.1 Galadriel sent1 Model

**Figure 6.15: The integrated Galadriel sent1 model with its features**

Similar to the model described in chapter 5, Galadriel sent1 is a simple sentiment model that calculates the sentiment score (senti-score value) of each word and adds up raw sentiment scores of all the words (calculating total) in the sentence/document, as follows:

**senti-score:** Multiplication of polarity and magnitude

\[
\text{Wordsenti-score} = \text{Wordpol} \times \text{Wordmag}
\]

**total:** Calculating total value of the document/sentence by adding senti-score value to previous word’s total value:

\[
\text{Wordtotal} = \text{Wordsenti-score} + \text{prev Wordtotal}
\]
In addition, I added the following features, which detect the contextual information of each lexical item in the given sentence/document:

- Detect whether the lexical item is in a negation or irrealis context (neg-context and block-context): getting the right values for neg-context and block-context of lexical items depends on their context (see chapter 4).

- Detect the context polarity (context-pol) of each lexical item in the sentence. This helps to assign polarity for context-dependent words in the next phase. I proposed the following rules, according to (Ding et al., 2008)'s linguistic conventions:

1. If the word is non-neutral and present in a negation sentence/phrase (the sentence contains a negation word), then the context polarity of the word is opposite to the polarity of the word. Otherwise, it takes the same polarity.

2. Context polarity of neutral words takes the previous word’s sentence polarity.

3. If the word *too* is present before any neutral adjectives (context-dependent words) in the sentence, then its context polarity is assigned as negative.

4. The words *however* and *but* flip the previous context polarity. However, if the word *but* is followed by *also*, it continues with the same sentence polarity.

To illustrate, consider the following example:

```
'this is a good phone but it ..'
```

<table>
<thead>
<tr>
<th>Word</th>
<th>pol</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>+1</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wordcon-pol</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+1</td>
<td>+1</td>
<td>-1</td>
<td>-1</td>
<td></td>
</tr>
</tbody>
</table>

The above example shows how the context polarity of each word is assigned using the above rules. The algorithm for assigning the *context-pol* values is as follows:

For any word in a sentence,

```java
if The word is polar word then

    if The word is in a the negation sentence then
        Wordcon-pol = Wordpol × − 1
    else
        Wordcon-pol = Wordpol
    end if

else

end if
```
if If previous word is too (but present word is not a boundary word or polar) then
    Word_{con-pol} = -1
else

    if Word = but or however then
        Word_{con-pol} = prev Word_{con-pol} × -1
    else
        Word_{con-pol} = prev Word_{con-pol}
    end if
end if

<sent1 context-pol> == < IFEQ: <here sent1 type .> positive
    THEN test-negcon1 ELSE test-neg .> >
<test-negcon1> == < IFEQ: <here sent1 neg-context .> yes
    THEN case nega ELSE case posi .> >
<case posi> == 1
<test-nega> == < IFEQ: <here sent1 type .> negative
    THEN test-negcon2 ELSE test-too .> >
<test-negcon2> == < IFEQ: <here sent1 neg-context .> yes
    THEN case posi ELSE case nega .> >
<case nega> == -1
<test-too> == < IFEQ: <here sent1 prev sent1 word .> too
    THEN test-lastword ELSE test-howevers .> >
<test-lastword> == < IFEQ: <here sent1 type .> boundary
    THEN case default ELSE <case nega>
<test-howevers> == < IFEQ: <here sent1 word .> however
    THEN case rule1 ELSE test-but .> >
<case rule1> == Eval:< here sent1 prev sent1 context-pol> * -1 .>
<test-but> == < IFEQ: <here sent1 word .> but
    THEN case rule2 ELSE case deafult .> >
<case rule2> == < IFEQ: <here sent1 next sent1 word .> also
    THEN case default ELSE case rule1 .> >
<case default> == <here sent1 prev sent1 context-pol>

Figure 6.16: Galadriel code for assigning context-pol values for words in a sentence in the sent1 model

An exception rule is applied when assigning a context-pol value for the word but. Generally, the context-pol value of the word but is opposite to its previous word’s context-pol, because but expresses the opposite polarity of the previous phrase/sentence, however the phrase but also does not express opposite polarity. This is similar to handling the phrase not only, as these two phrases come in the same sentence and express a different meaning to their lexical semantics.
The following exception rule was applied to assign a context-pol value for \textit{but}, when it is followed by \textit{also}:

For word \textit{but}

\begin{verbatim}
if  next Word = also then
    but\text{con-pol} = prev Word\text{con-pol}
else
    but\text{con-pol} = prev Word\text{con-pol} \times -1
end if
\end{verbatim}

In summary, polarity, magnitude and sentiment scores of each word in a given sentence/document are assigned in this phase. Then the total score of the whole sentence/document is calculated. This phase also detects negation and irrealis sentences. In addition, the contextual polarity of each word is also worked out using some rules. This can be explained by considering a simple example review from Ding et al.’s dataset\footnote{https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html}:

‘This is a good phone, but it has a small screen.’

In the above example, \textit{good} is an obvious positive sentiment lexical item or type of POSITIVE class, of which polarity and magnitude can be assigned directly from Galadriel’s sentiment dictionary. This makes the first half of the sentence positive. However, the \textit{but} switches the polarity and makes the rest of sentence negative. Figure 6.15 shows a brief explanation of the sent1 model’s features.

6.1.3.2 \textit{Galadriel sent2 Model}

Model sent2 is a model inherited from the sent1 model and is designed for handling context words (that inherit from the CONTEXT node), which express sentiment only in context. This model uses the context polarity of the lexical item, which is already assigned in sent1 for each word in the sentence. The context-pol value is used to identify its contextual polarity. This model is designed to assign polarity for context words using the following steps:

1. For clauses with a context word, the clause is checked if it is a negation clause.
2. For negation clauses, polarity of context word takes its opposite context polarity.
3. For non-negation clause, polarity of the context word shares its context polarity.

Consider the following example review:
The above sentence contains two clauses, joined by the conjunction *but*. The first clause has a positive sentiment word. Although the second clause does not have any polar words, it has a context word, *small*, which expresses negativity. The conjunction *but* changes the opinion of the first clause. Accordingly, the system can identify the contextual information/polarity of the context word *small* in the sentence. However, *but* does not behave in the same manner, when but comes with the word also. The phrase *but also* does not change the polarity or direction of its previous clause. For example:

*The food is very good, but also it comes in small portions.*

In the above example, the second clause does not express negativity. Hence, *small* does not express negative behaviour. In contrast, it expresses positivity. Figure 6.17 shows the rules used for handling context-dependent words.

The following algorithm is used in sent2; figure 6.18 shows the *Galadriel* code for sent2.

```plaintext
if Word = CONTEXT word then
    if Word_{neg-context} = yes then
        Word_{pol} = Word_{pol} \times con-pol - 1
    else
        ...
```

Figure 6.17: The features handle context dependent words in *Galadriel* model-2

*‘this is a good phone, but it has a small screen’*
Figure 6.18: The Galadriel code: the context-pol value is calculated in the sent2 model

\[
\text{Word}_{\text{pol}} = \text{Word}^2_{\text{con-pol}}
\]

end if
else
\[
\text{Word}_{\text{pol}} = \text{Word}^1_{\text{pol}}
\]
end if

6.1.3.3 Galadriel sent3 Model

![Diagram of Galadriel sent3 Model]

---

Figure 6.19: Intensifiers recalculate the magnitude values in Galadriel model 3

‘The movie was very nice’

The sent3 model is an extended version of sent2 that was designed for intensification. I adopted the SO-CAL rule to deal with intensifiers in order to calculate a word’s
sentiment magnitude. As discussed above, only intensifiers have the associated factor feature value, which modifies its neighbouring sentiment magnitude feature value. The integrated Galadriel model calculates only magnitude feature value in sent3, using the following equation:

For each word,

\[
\text{if } \text{prev Word} = \text{INTENSIFIER word} \text{ then } \\
\text{Word}_{\text{mag}} = \text{Word}^{2}_{\text{mag}} \times \text{prev Word}_{\text{fac}} \\
\text{else} \\
\text{Word}_{\text{mag}} = \text{Word}^{3}_{\text{mag}} \\
\text{end if}
\]

6.1.3.4 Galadriel sent4 Model

Model sent4 is another extended model, which has a special rule for calculating feature values for negation sentences (sentences with any negation words). Similar to the intensification, the integrated Galadriel sent4 model adopt the SO-CAL shift negation rule (figure 6.20 summarises the sent4 model). However, I do not use their constant number to shift the score. Instead, I reduce the magnitude value of the lexical item by 30% and switch its polarity value as follows:

\[
\text{if } \text{Word}_{\text{neg-context}} = \text{yes} \text{ then } \\
\text{Word}_{\text{pol}} = \text{Word}^{3}_{\text{pol}} \times -1;
\]

Figure 6.20: The negation rules are applied to the sent4 model followed by a negator
\[ \text{Word}_{\text{mag}} = \text{Word}_{\text{pol}}^9 \times 0.3 \]

else
\[ \text{Word}_{\text{pol}} = \text{Word}_{\text{pol}}^9 ; \]
\[ \text{Word}_{\text{mag}} = \text{Word}_{\text{mag}}^9 \]
end if

### 6.1.3.5 Galadriel sent5 Model

Figure 6.21: Irrealis rules are applied to words that come after an irrealis word in the model.

Another extended model, sent5 is designed to handle irrealis blocking, and has the same rules as the SO-CAL system. However, sent5 reassigns the polarity value to 0, but the magnitude value remains the same for words in the irrealis sentences. Figure 6.21 shows the rules for recalculating the feature values of an irrealis sentence in sent5.

### 6.1.3.6 Galadriel sent6 Model

The sent6 model handles interrogative sentences which are also irrealis sentences. However, identifying interrogative sentences is not easy. I used determination with question to identify interrogatives in chapter 5. However, the integrated Galadriel model handles this slightly differently, in order to give better performance.
Figure 6.22: The rules handling interrogative sentences in Galadriel model sent6

Interrogative sentences should start with a QUESTION lexical item and finish with a question mark. This model identifies if a given sentence is interrogative. Then it applies the similar rule of sent5. In this model, a rule was added to identify the ques-context and change the polarity value of any lexical item within the sentence to 0. Similar to the Galadriel features neg-context and block-context, this model uses ques-context to detect interrogation sentences.

As a default value, each lexical item has the value of no for their ques-context features. The following algorithm is used to assign the ques-context value in the sent6 model:

if The word is BOUNDARY word then
    Wordques-context = no
else

    if the word is a QUESTION word then

        if the next item is '?' then
            Wordques-context = yes
        else
            Wordques-context = the word after next Wordques-context
        end if
    else

    end if
Then the model uses the \texttt{ques-context} value of each lexical item in the sentence to assign the \texttt{polarity} value for the lexical item as follows:

\begin{verbatim}
if \texttt{Word\_ques-context} = \texttt{yes} then
  \texttt{Word\_ques-pol} = 0
else
  \texttt{Word\_ques-context} = \texttt{Word\_5\_ques-context}
end if
\end{verbatim}

6.2 Evaluation

The previous section explained the modelling of sentiment analysis using the inheritance mechanism by integrated \textit{Galadriel}. In chapter 5, I proposed a pre-evaluation method that sets cut-off values (boundary scores of \textit{Galadriel} scores) for sentiment classes. I tested the system for different sentiment analysis tasks, sentence-level and document-level.

This section presents the evaluation results for sentence- and document-level tasks. I used performance measures, precision, recall and f-score values, for the evaluation. I calculated precision and recall scores for each sentiment class and used their macro-average for the system’s total precision and recall. Then I calculated f-score (harmonic mean) of the overall system using the macro-average of precision and recall. I also present each class and the overall accuracy. I also discuss the datasets that were used for the evaluation.

6.2.1 Evaluation of the Integrated \textit{Galadriel} Model

I developed the final version of the complete \textit{Galadriel} system (the integrated system, \textit{Galadriel} version 1.0) by adding various techniques to the inherited models. I then performed separate evaluations against the original SO-CAL system. I used the same datasets that I employed in chapter 5 for the evaluation.

Table 6.1 shows f-scores of the integrated \textit{Galadriel} system on positive and negative reviews, compared with the basic \textit{Galadriel} system, which was discussed in chapter 5 (section 5.1.1), and the original SO-CAL system. The final f-scores demonstrate that the extended version of \textit{Galadriel} improved the system’s performance. The above
evaluations were performed based on binary (positive and negative) classification. The classes were classified based on the final Galadriel scores with calibrated method that I introduced in chapter 5. I also tested the integrated Galadriel system with various datasets on different levels of sentiment analysis tasks.

### 6.2.2 Sentence-Level Sentiment Analysis

This sentence-level sentiment analysis task for evaluation focused on identifying the author’s state of mind. That is to say, Galadriel detects if the author is in a positive, negative or neutral situation. For example, consider the following tweet;

‘Hay fever time is not good!’

The tweet shows the author’s negative sentiment. Moreover, the author is not expressing an opinion towards any product or entity. I used short tweets (one or two sentences) for the evaluation of the sentence-level task.

#### 6.2.2.1 Dataset

I used the STS-Gold dataset, which was presented by Saif et al. (2013) for tweeter sentiment evaluations. Saif et al. (2013) constructed the STS-Gold dataset from 180K tweets from the original Stanford Twitter corpus. They asked three graduate students to manually label the sentiment classes (positive, negative, neutral, mixed and other) using an instructed booklet. I used only the dataset that have been annotated with binary sentiment labels (632 positive tweets and 1402 negative tweets).

---

<table>
<thead>
<tr>
<th>Reviews</th>
<th>SO-CAL</th>
<th>Basic</th>
<th>Integrated</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pos-F</td>
<td>Neg-F</td>
<td>Pos-F</td>
</tr>
<tr>
<td>Books</td>
<td>0.69</td>
<td>0.74</td>
<td>0.76</td>
</tr>
<tr>
<td>Cars</td>
<td>0.90</td>
<td>0.89</td>
<td>0.92</td>
</tr>
<tr>
<td>Computers</td>
<td>0.94</td>
<td>0.94</td>
<td>0.95</td>
</tr>
<tr>
<td>Cookware</td>
<td>0.74</td>
<td>0.58</td>
<td>0.82</td>
</tr>
<tr>
<td>Hotels</td>
<td>0.76</td>
<td>0.67</td>
<td>0.79</td>
</tr>
<tr>
<td>Movies</td>
<td>0.84</td>
<td>0.84</td>
<td>0.91</td>
</tr>
<tr>
<td>Music</td>
<td>0.82</td>
<td>0.82</td>
<td>0.87</td>
</tr>
<tr>
<td>Phones</td>
<td>0.81</td>
<td>0.78</td>
<td>0.83</td>
</tr>
<tr>
<td>Total</td>
<td>0.81</td>
<td>0.79</td>
<td>0.85</td>
</tr>
</tbody>
</table>

Table 6.1: Comparison performance integrated Galadriel with SO-CAL feature on positive and negative reviews
The dataset has also been interpreted with targeted entities and polarities. However, I used only tweets for the evaluation of this task. I had Galadriel output scores ranging between $-14.53$ and $+26.79$. Then I also carried out the pre-evaluation process which was proposed in chapter 5 using a small test dataset (50 documents). I determined the following threshold for sentiment classes from the pre-evaluation process in order to carry out the calibrated evaluation:

\[
\text{Negative class := Galadriel score} < +1.95 \\
\text{Positive class := } +1.95 < \text{Galadriel score}
\]

I computed precision, recall and f-score of Galadriel system on STS-Gold dataset with both uncalibrated and calibrated method.

### 6.2.2.2 Evaluation Results

In order to compare the constructed STS-Gold dataset with the various datasets, Saif et al. (2013) used the Maximum Entropy (MaxEnt) classifier using the Mallet tool kit\(^\text{10}\) and carried out a binary sentiment classification on all the datasets.

Table 6.2 shows calculated precision, recall and f-scores of Galadriel using uncalibrated and calibrated evaluation methods, along with Saif et al. (2013)'s Maximum Entropy classifier. The results show the same recall for the positive datasets calculated by calibrated and uncalibrated methods. This is because positive documents had higher Galadriel scores and both methods produced the same ‘True positive’ value for positive datasets. However, the dataset is unbalanced and the uncalibrated method gives poor recall for the negative datasets and poor precision for the positive datasets. Overall, it shows the calibrated evaluation method in Galadriel leads to improved evaluation metrics. Moreover, similar to the Maximum Entropy classifier, Galadriel produces a better f-score on negative datasets. This is because, the dataset contains more negative tweets than positive tweets.

Saif et al. (2016) used the same STS-Gold dataset to evaluate their system SentiCircle, which is a lexical-based approach to Twitter sentiment analysis that handles contextual and conceptual (entity level) semantics of words. Like any other typical lexical-based approaches, SentiCircle also used the lexical negation rule and publicly available sentiment lexicons associated with sentiment orientation. In addition, SentiCircle captured contextual information of words in the tweets and updates

\(^{10}\text{http://mallet.cs.umass.edu/}\)
Table 6.2: Evaluation results of the Galadriel system on the STS-Gold dataset using both uncalibrated and calibrated evaluation methods along with the Maximum Entropy classifier used by [Saif et al.] (2013)

<table>
<thead>
<tr>
<th></th>
<th>Uncalibrated Method</th>
<th>Calibrated Method</th>
<th>MaxEnt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision Recall F-Score</td>
<td>Precision Recall F-Score</td>
<td>F-Score</td>
</tr>
<tr>
<td>Negative</td>
<td>0.9205 0.6776 0.7806</td>
<td>0.9343 0.8324 0.8804</td>
<td>0.8999</td>
</tr>
<tr>
<td>Positive</td>
<td>0.5489 0.8703 0.6732</td>
<td>0.7006 0.8703 0.7763</td>
<td>0.7490</td>
</tr>
<tr>
<td>Overall</td>
<td>0.7347 0.7739 0.7269</td>
<td>0.8175 0.8513 0.8284</td>
<td>0.8245</td>
</tr>
</tbody>
</table>

Table 6.2: Evaluation results of the Galadriel system on the STS-Gold dataset using both uncalibrated and calibrated evaluation methods along with the Maximum Entropy classifier used by [Saif et al.] (2013)

Table 6.3: Evaluation result comparison between Galadriel and other systems on STS-Gold dataset

I used [Saif et al.] (2016)’s best performance of the SentiCircle method, their baseline method and the Entropy classifier to compare the Galadriel performance. Table 6.3 shows the comparison of all the systems’ performance results. F-score of Galadriel calculated with the calibrated method shows outstanding results. The accuracy of the Maximum Entropy classifier is slightly better than Galadriel. However, for classification purposes, f-score is a more useful metric than accuracy, especially for an uneven dataset.
6.2.3 Evaluation for Idioms

I have showed that I included further techniques to the *Galadriel* model that cope with multi-word (two- or three-word) sentiments. In this way, I extended the *Galadriel* lexicon with sentiment idioms. However, the current *Galadriel* version simply finds exact word matches. I would try to improve *Galadriel* so that it uses stems instead of words in the future. I aimed to evaluate *Galadriel* in regards to how well it picks out idioms in a sentence. This task is also a sentence-level analysis task. I used simple sentences which express a sentiment using idioms. Consider the following example:

‘It seems to rise to the occasion.’

In the above sentence, though there is not positive words, *rise to the occasion* indicates positivity, the whole sentence expresses a positive sentiment.

6.2.3.1 Dataset

I used a list of idioms\footnote[11]{http://users.cs.cf.ac.uk/I.Spasic/idioment/} which were annotated by Williams et al. (2015) and modelled them in *Galadriel*, as shown in chapter 5. For the evaluation, I used Williams et al.’s dataset\footnote[12]{11} which has a list of sentences (2521) annotated with a sentiment label, positive (677), negative (1219) or other (neutral and ambiguous) (625), and each sentence contains an idiom. I also used the Williams et al.’s list of idioms which were collected various sources, such as educational websites and the British National Corpus, and annotated them using a web-based annotation platform. To validate the investigation of idioms, Williams et al. experimented with their system in regards to the idioms, via two different methods. They used SentiStrength (Thelwall et al., 2010) in their first experiment and Stanford CoreNLP’s sentiment annotator (Socher et al., 2013) for their second experiment as baseline methods and as part of the feature selection method. Finally, they used Weka (Hall et al., 2009) to train the classifier and perform the sentiment classification.

I carried out *Galadriel* system on the dataset. I had final *Galadriel* score output ranging between -17.52 and +14.65. The following cut-off values were obtained for the sentiment classes using a small dataset of annotated sentences (50 documents)
from the pre-evaluation process:

- Negative class := \( \text{Galadriel score} < -2.40 \)
- Other class := \(-0.45 < \text{Galadriel score} < +3.25 \)
- Positive class := \( +3.25 < \text{Galadriel score} \)

### 6.2.3.2 Evaluation Results

According to the sentiment class cut-off values, I calculated the performance measures and compared them with both baseline methods (Williams et al., 2015), used with SentiStrength and Stanford CoreNLP’s sentiment annotators, as shown in Table 6.4. Table 6.4 represents precision, recall and f-score values for each class and method separately. As (Williams et al., 2015) computed micro-averaged results for f-score values for overall systems, I presented the micro-average for f-score.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Performance measures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
</tr>
<tr>
<td>Positive</td>
<td>Baseline-SentiStrength</td>
</tr>
<tr>
<td></td>
<td>Baseline-Stanford CoreNLP</td>
</tr>
<tr>
<td></td>
<td>Galadriel - UnCalibrated Evaluation</td>
</tr>
<tr>
<td></td>
<td>Galadriel - Calibrated Evaluation</td>
</tr>
<tr>
<td>Negative</td>
<td>Baseline-SentiStrength</td>
</tr>
<tr>
<td></td>
<td>Baseline-Stanford CoreNLP</td>
</tr>
<tr>
<td></td>
<td>Galadriel - UnCalibrated Evaluation</td>
</tr>
<tr>
<td></td>
<td>Galadriel - Calibrated Evaluation</td>
</tr>
<tr>
<td>Others</td>
<td>Baseline-SentiStrength</td>
</tr>
<tr>
<td></td>
<td>Baseline-Stanford CoreNLP</td>
</tr>
<tr>
<td></td>
<td>Galadriel - UnCalibrated Evaluation</td>
</tr>
<tr>
<td></td>
<td>Galadriel - Calibrated Evaluation</td>
</tr>
<tr>
<td>Overall</td>
<td>Baseline-SentiStrength</td>
</tr>
<tr>
<td></td>
<td>Baseline-Stanford CoreNLP</td>
</tr>
<tr>
<td></td>
<td>Galadriel - UnCalibrated Evaluation</td>
</tr>
<tr>
<td></td>
<td>Galadriel - Calibrated Evaluation</td>
</tr>
</tbody>
</table>

Table 6.4: Evaluation result comparison between Galadriel with both calibrated and uncalibrated evaluation methods and the baseline methods

For all systems, evaluation metrics for other class shows comparatively poor results. Galadriel - uncalibrated method classifies the document as others, if its total Galadriel score is 0 and not many documents got Galadriel score of 0. Therefore, proportion of 'Tru positive' value for others and number of documents that have been classified as others is higher. This gives higher precision. Moreover, uncalibrated method assigns the positive label to the document with the total Galadriel
score is greater than 0. These are main reasons for overall poor performance of uncalibrated evaluation method. However, these issues have been overcome by Galadriel-calibrated evaluation method. Finally, Galadriel with uncalibrated method shows better performance than the other two baseline methods too.

6.2.4 Document-Level Sentiment Analysis

In the document-level task, I aimed to analyse the sentiment of a given text towards the topic of the document. In this task, Galadriel produces numeric scores similar to the sentence-level task. But the size of the output scores might be larger than the output scores of the sentence-level task. Then I calculate the boundary scores (or cut-off values) for the sentiment classes, as described in the pre-evaluation process in chapter 5.

6.2.4.1 Dataset

I used scale movie reviews\footnote{\url{http://www.cs.cornell.edu/people/pabo/movie-review-data/}} which were used by Pang and Lee\textsuperscript{13} for the evaluation of document-level sentiment analysis. Pang and Lee’s dataset contains four sets of movie reviews. A pair of authors reviewed each set. I used the set (1028 documents) reviewed by authors Dennis and Schwartz for this evaluation process. Pang and Lee employed SVM regression, SVM multiclass classification using one-vs-all (OVA) and metric labelling (a meta learning method) to address the rating-inference problem. The movie reviews are labelled in three classes (scale 0 (360 documents), 1 (427 documents), 2 (241 documents)) and four classes (scale 0 (172 documents), 1 (440 documents), 2 (302 documents), 3 (114 documents)). The movie reviews include a significant portion of description of the film, such as descriptions of its plot, actors, directors, etc. I removed those non-subjective sentences manually. Then I applied the Galadriel system to the movie review documents. As not all the movie review materials are the same length, I calculated a normalized score using the following equation:

\[
\text{Normalized Galadriel score} = \frac{\text{Total Galadriel score of the document}}{\text{Number of words in the document}}
\]

First I calculated evaluation metrics of Galadriel without using calibration method. As this is not a regular three class (positive, negative and neutral) classification, I cannot use the polarity sign to assign the scale. In order to classify the classes, I divided the ordinal Galadriel score range into three (for three scale) and four
(for four) classification classification. The documents with high Galadriel score has been assigned as the higher scale. Table 6.5 shows the computed evaluation result of Galadriel without the uncalibration method.

I then computed the cut-off values for each class for both labelling schemes. I used different datasets (the document set reviewed by James and Berardinelli) in the pre-evaluation process to compute cut-off values (see chapter 5 - calibration evaluation method). For the 3-class label, I had the following cut-off values:

\[
\begin{align*}
\text{scale} - 0 & := \text{Galadriel score} < -0.65 \\
\text{scale} - 1 & := -0.65 < \text{Galadriel score} < +1.05 \\
\text{scale} - 2 & := +1.05 < \text{Galadriel score}
\end{align*}
\]

And for the 4-class label, the following cut-off values were computed:

\[
\begin{align*}
\text{scale} - 0 & := \text{Galadriel score} < -2.05 \\
\text{scale} - 1 & := -2.05 < \text{Galadriel score} < -0.25 \\
\text{scale} - 1 & := -0.25 < \text{Galadriel score} < +1.55 \\
\text{scale} - 2 & := +1.55 < \text{Galadriel score}
\end{align*}
\]

6.2.4.2 Evaluation Results

The documents were classified into three and four groups of classes, according to computed cut-off values. Table 6.6 shows the performance measures of both three- and four-class classification in Galadriel. The mean average errors for three- and four-class classifications are 0.2706 and 0.3217. Moreover, the overall f-score of the three-class classification is better than the four-class classification in Galadriel. It shows the performance results of Galadriel using the calibration methods are much better.
Table 6.5: Evaluation of overall performance measures of three-class and four-class classification of *Galadriel* without the calibrated evaluation method

<table>
<thead>
<tr>
<th>Sentiment Classification</th>
<th>Performance measures</th>
<th>MAE</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
<td>F-Score</td>
</tr>
<tr>
<td>Three-class</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>scale-0</td>
<td>0.5881</td>
<td>0.6028</td>
<td>0.5953</td>
</tr>
<tr>
<td>scale-1</td>
<td>0.6735</td>
<td>0.4637</td>
<td>0.5492</td>
</tr>
<tr>
<td>scale-2</td>
<td>0.5370</td>
<td>0.8133</td>
<td>0.6469</td>
</tr>
<tr>
<td>Over all</td>
<td>0.5995</td>
<td>0.6266</td>
<td>0.6127</td>
</tr>
<tr>
<td>Four-class</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>scale-0</td>
<td>0.5345</td>
<td>0.3605</td>
<td>0.4306</td>
</tr>
<tr>
<td>scale-1</td>
<td>0.6026</td>
<td>0.5273</td>
<td>0.5624</td>
</tr>
<tr>
<td>scale-2</td>
<td>0.3343</td>
<td>0.3841</td>
<td>0.3575</td>
</tr>
<tr>
<td>scale-3</td>
<td>0.3500</td>
<td>0.5526</td>
<td>0.4286</td>
</tr>
<tr>
<td>Over all</td>
<td>0.4553</td>
<td>0.4561</td>
<td>0.4557</td>
</tr>
</tbody>
</table>

Table 6.6: Evaluation of overall performance measures of three-class and four-class classification of *Galadriel* with calibrated evaluation method

<table>
<thead>
<tr>
<th>Sentiment Classification</th>
<th>Performance measures</th>
<th>MAE</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
<td>F-Score</td>
</tr>
<tr>
<td>Three-class</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>scale-0</td>
<td>0.8049</td>
<td>0.7333</td>
<td>0.7674</td>
</tr>
<tr>
<td>scale-1</td>
<td>0.7808</td>
<td>0.8009</td>
<td>0.7908</td>
</tr>
<tr>
<td>scale-2</td>
<td>0.7481</td>
<td>0.8133</td>
<td>0.7793</td>
</tr>
<tr>
<td>Over all</td>
<td>0.7779</td>
<td>0.7825</td>
<td>0.7802</td>
</tr>
<tr>
<td>Four-class</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>scale-0</td>
<td>0.7239</td>
<td>0.5640</td>
<td>0.6340</td>
</tr>
<tr>
<td>scale-1</td>
<td>0.8186</td>
<td>0.8205</td>
<td>0.8195</td>
</tr>
<tr>
<td>scale-2</td>
<td>0.7217</td>
<td>0.7815</td>
<td>0.7504</td>
</tr>
<tr>
<td>scale-3</td>
<td>0.6587</td>
<td>0.7281</td>
<td>0.6917</td>
</tr>
<tr>
<td>Over all</td>
<td>0.7307</td>
<td>0.7235</td>
<td>0.7271</td>
</tr>
</tbody>
</table>

Table 6.7: Average accuracies comparison between *Galadriel* and Pang and Lee (2005)'s algorithms

<table>
<thead>
<tr>
<th>Sentiment Analysis System</th>
<th>Three-class</th>
<th>Four-class</th>
</tr>
</thead>
<tbody>
<tr>
<td>OVA-SVM</td>
<td>0.74</td>
<td>0.60</td>
</tr>
<tr>
<td>Regression-SVM</td>
<td>0.71</td>
<td>0.61</td>
</tr>
<tr>
<td>OVA-Metric labelling</td>
<td>0.73</td>
<td>0.63</td>
</tr>
<tr>
<td>Regression-Metric labelling</td>
<td>0.78</td>
<td>0.62</td>
</tr>
<tr>
<td><em>Galadriel</em></td>
<td>0.78</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Table 6.7 provides a comparison between the best performances of Pang and Lee.
(2005)'s algorithms and *Galadriel* with the calibration evaluation method. It shows
the *Galadriel* performance for three-class matches Pang and Lee (2005)'s regression
with metric labelling method’s best performance. *Galadriel* outperforms on four-
class classification.

### 6.3 Aspect-Based Sentiment Analysis

Unlike sentence-based and document-based sentiment analysis tasks, the aspect-

based task needs some additional work, which is modelling aspect lexical items and
aspect terms. A customised system has to be created for aspect-level sentiment
tasks, which is a base model with additional special cases in its lexicon, so the
inheritance architecture works well here.

#### 6.3.1 The *Galadriel* Base Model: Aspect Terms for Aspect-Based Sentiment Analysis

![Figure 6.23: Aspect class terms are structured in the hierarchy](image)

Aspect-based sentiment analysis is modelled in *Galadriel* by assuming that the lex-
icons for aspects and aspect terms are given by choosing a certain aspect. However,
aspects can be referred to by various aspect terms. Therefore, it is important to
identify the aspect terms that have been used in the customer reviews. I used a
training dataset to populate aspect terms, using Sketch Engine\textsuperscript{14}. For instance, a training set from the restaurant domain contains different types of customer reviews about aspects of the restaurant, such as *food, price, location, service*. These aspects can be extracted from the training dataset. In the reviews, different terms can be used to indicate the above aspects, such as *rice, pizza, burger*, for FOOD, *cheap, expensive* for PRICE, etc. These terms are also populated from the training dataset, which is already labelled with aspects. Some aspect terms may indicate sentiment, which has also been populated from training datasets labelled with a sentiment orientation. For example, *cheap* expresses a positive sentiment for the aspect PRICE, while *expensive* expresses a negative sentiment.

### 6.3.2 *Galadriel* sent7 Model

This model was designed for aspect-based sentiment level tasks that identify sentiment regarding specific given aspects. This task is similar to the task of the OO system. In previous models, the values of *magnitude* and *polarity* were recalculated by taking into account valence shifters, and then the final value of *senti-score* was calculated, and it was aggregated and calculated as *total* value, which gives the overall sentiment score/orientation of the given document. In this model, aspects were identified for every lexical item in the document (this is similar to what I did in chapter 5, section 5.3.1). Finally, the *senti-score* values of the lexical items are aggregated for its aspect. I assumed that aspects are given. The following steps were used identify sentiment for the given aspects:

1. **Step 1:** The document is broken up into sentences/phrases using punctuation rules. Every sentence and phrase may refer to different aspects.
2. **Step 2:** For every sentence, a given aspect is checked to see whether it is present in the sentence.
3. **Step 3:** If any aspects are present in the sentence, then the *senti-score* of each lexical item present in the sentence is assigned to the *senti-score* value of the aspect, and these are aggregated for its *total* value. I could also separately aggregate the polarity value of the words for the aspect. However, it is not necessary to calculate magnitude separately for the aspect.

Modelling of steps 2 and 3 was shown in chapter 5 in the section on modelling OO in *Galadriel*. However, the OO system does not consider breaking sentences/document into clauses. Therefore, OO aggregates the sentiment score of all sentiment words towards all aspects in the sentence. This is not an efficient method for long sentences or sentences containing multiple clauses. The integrated *Galadriel* model has extra

\textsuperscript{14}https://old.sketchengine.co.uk/open/
features that mean it is able to aggregate the sentiment score of sentiment words towards the relevant aspects by breaking the sentence into clauses. Consider the following example:

‘the service is excellent, and the food is delicious, but it is expensive.’

Assume the given aspects are price, food and service. Figure 6.24 shows the total sentiment score (+5) of the sentence, calculated in sent6 according to the appropriate rules and algorithms. In this model, the sentence is broken into three phrases:

1. ‘The service is excellent,’
2. ‘the food is delicious,’
3. ‘it is expensive.’

The model identifies that the first sentence contains the aspect service, the second sentence has the aspect term food and the aspect price is referred by third sentence. The sentiment score for each aspect takes the senti-score value of every lexical item in the sentence where the aspect is found. Then the total sentiment score for each aspect is obtained by aggregating the sentiment scores. Finally, this gives the total sentiment score of the aspects: service is +5, food is +4 and price is −4

<table>
<thead>
<tr>
<th></th>
<th>The</th>
<th>service</th>
<th>is</th>
<th>excellent</th>
<th>...</th>
<th>food</th>
<th>is</th>
<th>delicious</th>
<th>...</th>
<th>expensive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sent7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+4</td>
<td>0</td>
</tr>
<tr>
<td>service</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>food</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>price</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

‘the service is excellent, and the food is delicious, but it is expensive’

Figure 6.24: Total sentiment score for each of the targeted aspects is calculated

To calculate aspect-level sentiment score, the document first needs to be broken into sentences. Then, the given aspects are identified in every sentence. The sent7 model is added to the top of the Galadriel model sent6. Similar to the modelling of OO, for aspect-based sentiment analysis tasks, I add three additional Galadriel features (this is only used for aspect-level sentiment analysis tasks), found ASPECTi, senti-score ASPECTi, total ASPECTi, to the base model and calculate the value in the sent7 model, as shown in the OO modelling process (see section 5.3.1). An extra technique was added to get the real value for found ASPECTi by considering...
BOUNDARY words that can break the sentence into clauses as follows:

for every word,

if $\text{Word} = \text{BOUNDARY word}$ then

$\text{Word}^\text{ASPECT}_f = \text{fail}$

else

if $\text{Word}^\text{ASPECT}_f = \text{true}$ or $\text{Word}^\text{ASPECT}_f = \text{true}$ then

$\text{Word}^\text{ASPECT}_f = \text{true}$

else

$\text{Word}^\text{ASPECT}_f = \text{fail}$

end if

end if

The Galadriel sent6 model calculates the senti-score value of each lexical item in the sentence. Then each lexical item’s targeted aspects are identified by breaking down the document into clauses, as in the Galadriel sent7 model. Then, it assigns the word’s senti-score value to its targeted ASPECT’s senti-score. Figure 6.24 shows that the total sentiment score of the sentence towards SERVICE, FOOD and PRICE are +5, +6 and −4, respectively.

6.4 Evaluation: Aspect-Based Sentiment Analysis

Aspect-level sentiment analysis tasks allow for extracting sentiment (whether positive, negative or neutral) regarding a particular aspect of the product. As I have already described, I do not focus on detecting the aspects. I process the aspect-level sentiment analysis task while assuming the aspects (terms) have been given.

This section discusses the performance of integrated Galadriel on an aspect task. First I compare the Galadriel sent7 model with the OO system. I used the same dataset used in section 5.4 and compared the results with the basic Galadriel system and the original OO system.

Table 6.8 shows that the integrated version of Galadriel performs better than basic Galadriel and the OO system. I also compared the integrated Galadriel system on SemEval2016 aspect-based sentiment analysis (ABSA) task.
Table 6.8: Comparison performance of integrated Galadriel and basic Galadriel with OO features.

<table>
<thead>
<tr>
<th>Product name</th>
<th>OO</th>
<th>B. Galadriel</th>
<th>Int. Galadriel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
</tr>
<tr>
<td>Digital camera1</td>
<td>0.93</td>
<td>0.92</td>
<td>0.93</td>
</tr>
<tr>
<td>Digital camera2</td>
<td>0.96</td>
<td>0.96</td>
<td>0.96</td>
</tr>
<tr>
<td>Cellular phone1</td>
<td>0.93</td>
<td>0.9</td>
<td>0.91</td>
</tr>
<tr>
<td>MP3 player</td>
<td>0.87</td>
<td>0.86</td>
<td>0.87</td>
</tr>
<tr>
<td>DVD player</td>
<td>0.89</td>
<td>0.88</td>
<td>0.89</td>
</tr>
<tr>
<td>Cellular phone2</td>
<td>0.95</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>Router</td>
<td>0.84</td>
<td>0.83</td>
<td>0.83</td>
</tr>
<tr>
<td>Antivirus software</td>
<td>0.9</td>
<td>0.88</td>
<td>0.88</td>
</tr>
<tr>
<td>Total</td>
<td>0.91</td>
<td>0.90</td>
<td>0.90</td>
</tr>
</tbody>
</table>

6.4.1 Dataset

I used a dataset[^1] which was used in task 5, SemEval2016 by Pontiki et al. (2016), for the evaluation of this task. I needed an annotated dataset; I used their English training datasets of restaurant and laptop domains. In the original SemEval2016 task-5, there were two subtasks. Subtask 1 involved sentence-level analysis and had three slots:

1. Aspect category
2. Opinion target expression
3. Sentiment polarity

Subtask 2 was a text-level (or document-level) analysis task, and its output was to detect pairs of aspect and polarity. This task is more similar to Pontiki et al. (2016)’s subtask 2, as it is a document-level task and more focused on slot 3 of their subtask 1, while assuming the slot 1 task has been performed. Therefore, I extracted the targeted aspects using Sketch Engine and manually assigned them for each domain:

**Restaurant:** service, place, food, price, drinks, ambience, general

**Laptop:** screen, battery, price, software, display, hard disk, key board, quality, operation performance, company, design/feature, support, multi device, usability, portability, connectivity, memory, CPU, OS, shipping, graphics, hardware, ports, mouse

For aspect-based sentiment analysis tasks, I had to create customised systems for the evaluation to incorporate the aspect information. I did not focus on separate

entities or attributes for aspects, as in Pontiki et al. (2016) work. Instead, I collected a set of lexical entries that are used to express any of the above targeted aspects. I also collected synonyms from various web dictionaries, and the domain-specific sentiment lexicon for each aspect, and modelled them in Galadriel. Moreover, a part of the training dataset was used to compute the cut-off values for sentiment classes. I manually selected the dataset for the pre-evaluation and obtained cut-off values for the sentiment classes. I used a mixture of documents from both domains to set the cut-off values in the pre-evaluation process. The same cut-off values were used for both documents, as the documents of both domains have a similar range of sentiment scores.

\[
\text{Negative} := \text{Galadriel score} < -0.75 \\
\text{Neutral} := -0.75 < \text{Galadriel score} < +1.25 \\
\text{Positive} := +1.25 < \text{Galadriel score}
\]

I tested the Galadriel system (Galadriel sent7 model) with the test dataset\textsuperscript{16} comparing the results with its annotated results.

### 6.4.2 Evaluation Results

This evaluation method was slightly different from the SemEval2016 assessment. This aspect-level sentiment task involved the slot 2 and slot 3 tasks of SemEval2016. Aspect terms and sentiments were identified in this task. I considered only the correct pair of ‘aspect – sentiment (positive/negative/neutral)’ which was correctly classified by Galadriel (‘true positive’ values).

I used precision, recall and f-score values for this evaluation, whereas SemEval2016 (Pontiki et al., 2016) used accuracy for the evaluation of slot 3, and f-score for slot 2. Therefore, I also calculated the accuracy of overall Galadriel output in order to compare my results with the participants of SemEval2016.

As mentioned above, I used positive (TP) values to calculate precision, recall and f-score values for each aspect, as the dataset is unbalanced and a small number of documents were present in the neutral class. The uncalibration method classifies documents that have a Galadriel score of 0 as neutral. Only a small number of documents were classified as neutral. Therefore, precision for the neutral class in the uncalibration method gives a high value.(see table\textsuperscript{6.9})

<table>
<thead>
<tr>
<th>Domains</th>
<th>Sentiment Labels</th>
<th>\textit{Galadriel} Uncalibration Method</th>
<th>\textit{Galadriel} Calibration Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>P R F-Score</td>
<td>P R F-Score</td>
</tr>
<tr>
<td></td>
<td>Positive</td>
<td>0.9336 0.6840 0.7896</td>
<td>0.9777 0.9132 0.9443</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>0.6364 0.3043 0.4118</td>
<td>0.3396 0.7826 0.4737</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>0.4514 0.9186 0.6054</td>
<td>0.9867 0.8605 0.9193</td>
</tr>
<tr>
<td></td>
<td>Overall</td>
<td>0.6738 0.6357 0.6542</td>
<td>0.7680 0.8521 0.8079</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Positive</td>
<td>0.8360 0.7794 0.8067</td>
<td>0.9497 0.8882 0.9179</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>0.7500 0.1935 0.3077</td>
<td>0.2692 0.6774 0.3853</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>0.6114 0.7818 0.6862</td>
<td>0.9214 0.7818 0.8459</td>
</tr>
<tr>
<td></td>
<td>Overall</td>
<td>0.7324 0.5849 0.6504</td>
<td>0.7134 0.7825 0.7464</td>
</tr>
</tbody>
</table>

Table 6.9: Evaluation results of \textit{Galadriel}'s aspect-base model (sent7) using calibrated and uncalibrated methods.

<table>
<thead>
<tr>
<th>System</th>
<th>Domain</th>
<th>F-Score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{Galadriel} -Calibration method</td>
<td>Restaurant</td>
<td>0.8079</td>
<td>0.8942</td>
</tr>
<tr>
<td></td>
<td>Laptop</td>
<td>0.7464</td>
<td>0.8433</td>
</tr>
<tr>
<td>UWB/C</td>
<td>Restaurant</td>
<td>0.8096</td>
<td>0.8094</td>
</tr>
<tr>
<td></td>
<td>Laptop</td>
<td>0.6045</td>
<td>0.7449</td>
</tr>
<tr>
<td>UWB/U</td>
<td>Restaurant</td>
<td>0.8016</td>
<td>0.8193</td>
</tr>
<tr>
<td></td>
<td>Laptop</td>
<td>0.5972</td>
<td>0.7549</td>
</tr>
<tr>
<td>basel./C</td>
<td>Restaurant</td>
<td>0.7871</td>
<td>0.7425</td>
</tr>
<tr>
<td>bunji/U</td>
<td>Laptop</td>
<td>0.5268</td>
<td>0.7303</td>
</tr>
<tr>
<td></td>
<td>Restaurant</td>
<td>0.7978</td>
<td>0.7054</td>
</tr>
<tr>
<td></td>
<td>Laptop</td>
<td>0.5472</td>
<td>0.6000</td>
</tr>
</tbody>
</table>

Table 6.10: Comparison f-score and accuracies between \textit{Galadriel} and some participations in SemEval2016 on Restaurant and Laptop Domains.
Table 6.10 shows some of the best performances of teams who participated in SemEval2016 for ABSA, along with Galadriel’s results.

## 6.5 Neutral Model: An Extended System

The neutral class has not been given as much attention as the positive and negative classes, as most researchers focus on binary classification. However, the neutral class is not neglectable, and it does impact on the performance of a sentiment analysis system. Although machine learning approaches detect the neutral class by labelled training datasets, lexicon-based approaches indicate a document is in the neutral class if its final semantic score is 0, which is not always true. A document should belong to the neutral class if it is an objective sentence or if it contains only non-polar words.

I thus introduced an extended model that detects neutral class documents. In order to model this in Galadriel, I introduce two features, tot-pos and tot-neg, which replace the total feature. The tot-pos and tot-neg features calculate total positive sentiment score and negative sentiment score independently. Then neutral class documents can be identified if both tot-pos and neg-pos values are 0.

The previous section discussed the implementation of developed extensions in building the final Galadriel system. Moreover, I showed that the Galadriel system produces a numeric sentiment score as its final output. Hence, the overall sentiment polarity is decided by the sign of the final score. According to the definition, a neutral review/document/sentence is one which produces a Galadriel score of 0. However, this is not always true. Consider the following example:

*The room was acceptable but expensive.*

If the above sentence is analysed for the aspect-based task, then the output would be as follows: the sentiment score of the phrase is +1 (positive; using Liu et al.’s (2008) lexicon) towards the aspect *place*, and –1 (negative) towards the aspect *price*. However, if it is analysed at the document level, the final score would be 0 which expresses that the sentiment is neutral, even though it is not neutral.
In this section, I propose a definition of the neutral class and discuss modelling this definition in *Galadriel*. First, I defined a sentence/document as neutral, only if it does not contain any polar (positive or negative) words. I made some changes to the features in *Galadriel*’s models to identify neutral language. I added two additional features to the SENTIMENT node, instead of the total feature. They are:

- **tot-pos**: This feature refers to the total positive sentiment score of the sentence, which is calculated by aggregating the *senti-score* value of all positive lexical items in the sentence.

- **tot-neg**: This feature value is calculated by aggregating the *senti-score* value of all negative lexical items in the sentence.

In the base model, I added the default value 0 to the above features. Then I added the following rule to the sent 1 model to calculate the value of the *tot-pos* and *tot-neg* features. Then the rules and values are inherited to the other the other *Galadriel* models.

\[
\text{if } \text{Word}_{\text{senti-score}} > 0 \text{ then} \quad \text{Word}_{\text{tot-pos}} = \text{Word}_{\text{senti-score}} + \text{prev } \text{Word}_{\text{tot-pos}} \\
\text{else} \quad \text{Word}_{\text{tot-pos}} = \text{prev } \text{Word}_{\text{tot-pos}} \\
\text{end if} \\
\text{if } \text{Word}_{\text{senti-score}} < 0 \text{ then} \quad \text{Word}_{\text{tot-neg}} = \text{Word}_{\text{senti-score}} + \text{prev } \text{Word}_{\text{tot-neg}}
\]
Figure 6.26: The rules are added to model 1 for the neutral class detection

Figure 6.26 shows Galadriel codes that were added to the Galadriel models to detect neutral language, and they produce output texts similar to figure 6.25. In the example output text, \textit{tot-pos} and \textit{tot-neg} values are 7 and \(-2.6\), which means the document/sentence expresses a mixed opinion. However, it actually expresses more positive view. Hence, I define a text as neutral only if both \textit{tot-pos} and \textit{tot-neg} values are 0. This model works better for irrealis blocking and interrogation phrases as well, because the lexical items within the irrealis blocking and interrogation sentences are identified in models sent5 and sent6, and their polarity values are switched to 0.

I tested the model in a sentence-level analysis, as most of the documents contain mixed sentiments rather than only neutral. The evaluation showed better performance results. Therefore, I have proved that Galadriel can detect unmixed neutral attitude sentences. However, I do not add this model to the primary Galadriel system because I used various datasets, which contain large documents with mixed opinions.

### 6.6 Evaluation of Neutral Class Classification

As an extended model, I thus proposed a new model for detecting neutral language in a sentence. This neutral model produces a pair of output \textit{senti-score} values which
contain positive and negative scores separately. I defined a sentence as neutral if both its *Galadriel* positive and negative scores are 0. I did not use the pre-evaluation process to evaluate the *Galadriel* neutral model because the calibration method in pre-evaluation is used for setting the threshold for sentiment classes, which is not necessary for the evaluation of the *Galadriel* neutral model.

### 6.6.0.1 Dataset

I used a human-coded text, which was created by Thelwall et al. (2010) in order to develop SentiStrength. Thelwall et al. used three independent coders to annotate six datasets of tweets and to train the classifier. The coders were asked to assign a sentiment, positive or negative, on a scale ranging from 1 to 5, where 1 indicates no sentiment and 5 indicates a strong positive or negative sentiment.

I used only four datasets for the evaluation, which were comments from MySpace, the BBC, Twitter and YouTube. Each dataset contains nearly 1000-4000 tweets/short informal pieces of text, and these were annotated with pairs of mean positive and mean negative values. The mean positive and negative values were calculated by taking the average of the score from all six human annotators. Moreover, I assumed any texts that have both mean positive value and mean negative values of 1 belong to the neutral class, and others are non-neutral.

### 6.6.0.2 Evaluation Results

I considered a sentence as neutral only if it is annotated with mean positive 1 and mean negative 1. The rest of the sentences are reviewed as non-neutral sentences. I experimented with the *Galadriel* system with the human annotated documents from (Thelwall et al. 2010)’s datasets. I tested the datasets with the *Galadriel* system twice. First, I tested the *Galadriel* neutral model with the datasets. Then, *Galadriel* without the neutral model (standard model) was tested with the same datasets. In these experiments, I only aimed to classify the neutral sentences. Please note that the calibration method was not used to set the cut-off value for the neutral class. I assumed any documents with a total *Galadriel* score of 0 were neutral sentiment-class documents. Both tests’ results were compared with the actual results. Then I computed precision, recall and f-score values; a comparison of both experiments is shown in table 6.11.

I defined a sentence as neutral only if the sentence contains non-sentiment words. The *Galadriel* neutral model produces positive and negative output scores separately, which allows us to identify neutral sentences according to my definition.
Table 6.11: Evaluation result comparison between the Galadriel neutral model and the Galadriel standard model on various domains

Table 6.11 shows that the Galadriel neutral model performs much better than the Galadriel standard system without the neutral model on all domains.

6.7 Discussion: The Integrated Model

The final Galadriel 1.0 system is a development of models with various rules and techniques that allow for handling of the major linguistic features of the lexical items as well as the irregular behaviour of lexical entries.

According to [Ding et al., 2008] method, a product is an object which has attributes (aspect or product features), components and sub-components. To make things simple, I did not consider these different levels. I only reviewed the product and its aspects. I collected the terms of components and sub-components of the aspects and modelled them under the appropriate aspect nodes in the Galadriel lexicon. For example, smartphone is an object or product, and speaker, screen and camera are its aspects (attributes). The word picture (component) was modelled under the CAMERA node. If a review talks about picture quality, then Galadriel identifies that the review is about the ‘Camera’.

I adapted [Ding et al., 2008] linguistics conventions to compute context-dependent opinion words. Their pseudo intra-sentence rule uses the conjunction ‘which’ to detect the sentiment of the nearest context-dependent word. Consider this example:

‘The camera has long battery life, which is great.’
long in the above sentence is identified as positive by Liu et al.’s OO system. But the Galadriel model is not able to compute the sentiment score using the Galadriel sent2 model, because the rules of the Galadriel sent2 model direct long to check only its previous lexical items. However, I could make ‘,’ a non-boundary item if its next word is which. This then makes the whole sentence one clause. Therefore, the positive lexical item, great, shares its senti-score with the aspect battery.

Sentiment analysis of comparative sentences is a key challenge. Non-equal gradable comparative sentences express that one object (Object1) is better than the other (Object2) by using the conjunction ‘than’, which was modelled as a BOUNDARY word in the Galadriel lexicon, so that it splits a sentence into two clauses. Consider the following example:

‘An iPhone is better than a Samsung phone.’

In the aspect-level sentiment task, let’s assume iPhone and Samsung are aspects. The lexical items better and iPhone are in the same clause. Therefore, the senti-score of iPhone takes the senti-score of better, whereas the senti-score of Samsung does not. In the document-level analysis, the sentence expresses a positive sentiment. So if the document is about an iPhone, then the Galadriel outcome is correct. On the other hand, if the review is about the Samsung phone, then the Galadriel result would be incorrect. This is one of the drawbacks of Galadriel, which is further discussed in chapter 7.

Modelling techniques for anaphora and cataphora are useful for the aspect-level sentiment analysis task. It is easy for co-references to access the information of their neighbouring items and behave accordingly in Galadriel. This process helps to detect the targeted aspects in aspect-level sentiment analysis tasks. I use the list of aspect words to identify the aspect in a sentence. In the document-level analysis, Galadriel calculates overall sentiment score for the given product. Therefore, I do not add any algorithms for anaphora and cataphora in the current Galadriel version.
6.8 Parametric Feature of *Galadriel*

This section discusses the main factors (parameters) that control and affect the *Galadriel* system and its performance measures. Each lexical item of the *Galadriel* lexicon has different properties (or sentiment behaviour), which are shared by groups of lexical items. I explained the properties by ‘feature:value’ pairs in previous chapters. Lexical items are structured (inheritance-based) in the *Galadriel* lexicon based on their feature:value pairs using the DATR mechanism. In the inheritance lexicon network, every node describes features by their value, which passes down to their subclasses. The values of *Galadriel* feature the factors that modify the final *Galadriel* scores. Therefore, the default feature values can be adjusted in order to improve the performance of the system or to get a different final score. **polarity** and **magnitude** are the main features that define the sentiment of a lexical item, as the other features do not directly affect the sentiment behaviour of lexical items. But the ‘polarity’ value of a lexical item is unique and only the default value of **magnitude** of lexical items is adjustable. Therefore, **magnitude** can be defined as *Galadriel*’s parametric feature.

As explained in previous chapters, I mainly use Taboada et al. (2011)’s dictionaries for the *Galadriel* lexicon, which contain a list of sentiment words with a score ranging from −5 for extremely negative to +5 for extremely positive, with an equal interval. For the *Galadriel* lexicon, separate nodes were created for the features **polarity** (sign) and **magnitude** (score value), and modelled in an inheritance structure (see figure 6.27). This section examines the qualitative interpretation of *Galadriel*’s parametric feature, which is the **magnitude** feature. To make the experiment
easy and effective, I slightly altered the levels of *Galadriel*’s lexicon inheritance. I switched the levels of *polarity* and *magnitude*, as shown in figure 6.28.

In the revised structure of the inheritance lexicon, five nodes describe the magnitude feature, with values of 1, 2, 3, 4 and 5 (I call them magnitude nodes), and which are inherited from SENTIMENT node. Then the information of the nodes passes down to their sub-nodes that describe the polarity feature (see figure 6.28).

---

Figure 6.28: Nodes of the inheritance structure describing magnitude nodes

To examine the qualitative measure of the parametric feature of *Galadriel*, I test the sensitivity and stability of the magnitude values of lexical items. Sensitivity tests how sensitive the *Galadriel* system is when small changes are made in the magnitude values of lexical items. Stability tests how stable the *Galadriel* system is when changes are made in the intervals between the nodes that describe magnitude.

### 6.8.1 Sensitivity

The revised *Galadriel* lexicon structure contains five separate nodes that describe the magnitude values 1, 2, 3, 4 and 5, which are inherited from the SENTIMENT node. I tested the system performance for slight changes in the magnitude values. In section 5.6.1 I described a calibration method to compute cut-off values (boundary scores) of classes in the pre-evaluation process. In this section, I test and compare *Galadriel*’s performance by changing the magnitude values of lexical items.
6.8.1.1 Experiment and Results

*Galadriel’s* DATR mechanism does allow for capturing generalisation and avoiding the duplication properties of the lexical items. Therefore, in order to make slight changes in the sentiment scores of each lexical item, the nodes describing the feature magnitude values were moved by $\pm 0.25, \pm 0.5, \pm 0.75$ and $\pm 1$, and I computed precision, recall and f-score values, with the computed cut-off values, using the calibration method. Figure 6.28 shows that the nodes describing the magnitude feature that are inherited from the SENTIMENT node have standard magnitude values (let’s say $n$).

I used Pang and Lee (2005) three-class classification dataset to test the sensitivity of *Galadriel’s* parametric feature (magnitude). I used a total of 150 movie reviews (authors: James and Berardinelli) and 50 reviews from each class (scale-0, scale-1, scale-2). The cut-off value of classes ($(-0.65$ for $C_{0/1}$ and $+1.05$ for $C_{1/2}$) were calculated in the pre-evaluation process (see section 5.6.1). The experiment was carried out with *Galadriel*. Then the final *Galadriel* score of each document was normalised and classified into three classes according to the cut-off values. The precision, recall and f-score values for each class and overall performance measures were calculated.

I then repeated the experiment, while changing the magnitude values. The nodes describing magnitude were changed (added and subtracted) by $0.25, 0.5, 0.75$ and $1$. However, instead of moving the magnitude value by $-1$, I moved it by only $-0.95$, in order to avoid the node VERY WEAK taking a magnitude value 0, as any weak positive and negative lexical items should not be assigned as neutral.

<table>
<thead>
<tr>
<th>Ex</th>
<th>Nodes</th>
<th>Values of magnitude nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>VERY-WK</td>
</tr>
<tr>
<td>1</td>
<td>$n - 0.95$</td>
<td>0.05</td>
</tr>
<tr>
<td>2</td>
<td>$n - 0.75$</td>
<td>0.25</td>
</tr>
<tr>
<td>3</td>
<td>$n - 0.50$</td>
<td>0.50</td>
</tr>
<tr>
<td>4</td>
<td>$n - 0.25$</td>
<td>0.75</td>
</tr>
<tr>
<td>5</td>
<td>$n$</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>$n + 0.25$</td>
<td>1.25</td>
</tr>
<tr>
<td>7</td>
<td>$n + 0.50$</td>
<td>1.50</td>
</tr>
<tr>
<td>8</td>
<td>$n + 0.75$</td>
<td>1.75</td>
</tr>
<tr>
<td>9</td>
<td>$n + 1$</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 6.12: Experiments carried out while changing the magnitude values of the *Galadriel* system

Table 6.12 shows the nine experiments that were carried out with the same dataset while moving the magnitude values of the nodes VERY WEAK to VERY STRONG.
by making small changes up to 1. Table \ref{6.12} shows that experiment 5 was carried out using the standard values of the magnitude nodes \((n)\) of the \textit{Galadriel} lexicon. Table \ref{6.13} shows precision, recall and f-score values for each class separately. Table \ref{6.14} provides macro-average precision, recall and f-score values for the three classes. Figure \ref{6.29} shows appropriate graphs according to the performance measures.

For both scale-2 \textit{(positive)} and scale-0 \textit{(negative)} classes, there were no changes in the performance measures until the values of magnitude changed by \(-0.25\). The recall for these classes did not change while the magnitude values increase. On the other hand, the recall for the \textit{scale-1} class decreased, and it increased while the value of magnitude decreased. The precision for \textit{scale-2} and \textit{scale-0} decreased when the magnitude values were changed by more than 0.25. All the performances remain the same for the whole \textit{scale-1} class, until the value of magnitude changed by more than 0.75. Graph \ref{6.29d} shows an overall macro-average of precision, recall and f-score values, which remain the same only when magnitude values are reduced by 0.25. I can thus conclude that the performance of \textit{Galadriel} does not change when the magnitude values of lexical items are reduced by up to 0.25.
### Table 6.13: Performance measures of three classes according to the different magnitude values of nodes

<table>
<thead>
<tr>
<th>Nodes</th>
<th>scale-0</th>
<th></th>
<th>scale-1</th>
<th></th>
<th>scale-2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>$n - 0.95$</td>
<td>0.7143</td>
<td>0.5000</td>
<td>0.5882</td>
<td></td>
<td>0.5625</td>
<td>0.9000</td>
</tr>
<tr>
<td>$n - 0.75$</td>
<td>0.7000</td>
<td>0.7000</td>
<td>0.7000</td>
<td></td>
<td>0.7273</td>
<td>0.8000</td>
</tr>
<tr>
<td>$n - 0.50$</td>
<td>0.7000</td>
<td>0.7000</td>
<td>0.7000</td>
<td></td>
<td>0.7273</td>
<td>0.8000</td>
</tr>
<tr>
<td>$n - 0.25$</td>
<td>0.7273</td>
<td>0.8000</td>
<td>0.7619</td>
<td></td>
<td>0.8000</td>
<td>0.8000</td>
</tr>
<tr>
<td>$n$</td>
<td>0.7273</td>
<td>0.8000</td>
<td>0.7619</td>
<td></td>
<td>0.8000</td>
<td>0.8000</td>
</tr>
<tr>
<td>$n + 0.25$</td>
<td>0.5333</td>
<td>0.8000</td>
<td>0.6400</td>
<td></td>
<td>0.6667</td>
<td>0.4000</td>
</tr>
<tr>
<td>$n + 0.50$</td>
<td>0.5333</td>
<td>0.8000</td>
<td>0.6400</td>
<td></td>
<td>0.6667</td>
<td>0.4000</td>
</tr>
<tr>
<td>$n + 0.75$</td>
<td>0.5333</td>
<td>0.8000</td>
<td>0.6400</td>
<td></td>
<td>0.6667</td>
<td>0.4000</td>
</tr>
<tr>
<td>$n + 1$</td>
<td>0.4000</td>
<td>0.8000</td>
<td>0.5333</td>
<td></td>
<td>0.6667</td>
<td>0.4000</td>
</tr>
</tbody>
</table>

### Table 6.14: Performance measures of the three classes according to the different magnitude values of nodes

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n - 0.95$</td>
<td>0.6637</td>
<td>0.6333</td>
<td>0.6481</td>
</tr>
<tr>
<td>$n - 0.75$</td>
<td>0.7350</td>
<td>0.7333</td>
<td>0.7341</td>
</tr>
<tr>
<td>$n - 0.50$</td>
<td>0.7350</td>
<td>0.7333</td>
<td>0.7341</td>
</tr>
<tr>
<td>$n - 0.25$</td>
<td>0.7684</td>
<td>0.7667</td>
<td>0.7675</td>
</tr>
<tr>
<td>$n$</td>
<td>0.7684</td>
<td>0.7667</td>
<td>0.7675</td>
</tr>
<tr>
<td>$n + 0.25$</td>
<td>0.6593</td>
<td>0.6333</td>
<td>0.6460</td>
</tr>
<tr>
<td>$n + 0.50$</td>
<td>0.6593</td>
<td>0.6333</td>
<td>0.6460</td>
</tr>
<tr>
<td>$n + 0.75$</td>
<td>0.6593</td>
<td>0.6333</td>
<td>0.6460</td>
</tr>
<tr>
<td>$n + 1$</td>
<td>0.5222</td>
<td>0.4667</td>
<td>0.4929</td>
</tr>
</tbody>
</table>
Figure 6.29: Graphs for performance measures of the three classes and overall measures against different magnitude values of nodes
6.8.2 Stability

In the previous section, I discussed the sensitivity of the parametric feature of Galadriel (magnitude). I tested the performance measures on the computed cut-off values while moving the values of magnitude nodes in a linear direction with equal intervals. In this section, I test for stability of the magnitude feature by testing Galadriel’s performance while changing the spread between the magnitude values of VERY WEAK, WEAK, MEDIUM, STRONG and VERY STRONG lexical items. Stability was tested in two ways:

1. The values of magnitude nodes in arithmetic sequences:
   A number of experiments were carried out with the values of magnitude nodes in five different arithmetic sequences. The intervals between nodes were equal for all five experiments. For instance, say that the standard values of magnitude nodes are \( n \), \( n = 1, 2, 3, 4, 5 \) for the nodes VERY WEAK, WEAK, MEDIUM, STRONG and VERY STRONG, respectively. Five different experiments were carried out for \( mn \), for \( m = 1, 2, 3, 4, 5 \) (see table 6.15).

<table>
<thead>
<tr>
<th>Ex</th>
<th>Nodes</th>
<th>Values of Magnitude Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( n )</td>
<td>( 1 ) ( 2 ) ( 3 ) ( 4 ) ( 5 )</td>
</tr>
<tr>
<td>2</td>
<td>( 2n )</td>
<td>( 2 ) ( 4 ) ( 6 ) ( 8 ) ( 10 )</td>
</tr>
<tr>
<td>3</td>
<td>( 3n )</td>
<td>( 3 ) ( 6 ) ( 9 ) ( 12 ) ( 15 )</td>
</tr>
<tr>
<td>4</td>
<td>( 4n )</td>
<td>( 4 ) ( 8 ) ( 12 ) ( 16 ) ( 20 )</td>
</tr>
<tr>
<td>5</td>
<td>( 5n )</td>
<td>( 5 ) ( 10 ) ( 15 ) ( 20 ) ( 25 )</td>
</tr>
</tbody>
</table>

Table 6.15: Experiments were carried out while changing the magnitude values of the Galadriel system in five arithmetic sequences

2. The values of magnitude nodes in polynomial sequences:
   Similar to arithmetic sequences, another set of experiments were carried out for five different values of magnitude nodes in five different polynomial sequences \( n^m \) for \( m = 1, 2, 3, 4, 5 \). Thus, the intervals between the nodes were different (see table 6.16).

6.8.2.1 Experiment and Results

To test the stability of the Galadriel’s parametric feature (magnitude), I used Pang and Lee (2005) three-class and four-class classification dataset\(^\text{17}\). I used 120 movie reviews (authors: Scott and Renshaw) and carried out the experiments in Galadriel with various values of magnitude nodes. Tables 6.15 and 6.16 show the computed

\(^{17}\)http://www.cs.cornell.edu/people/pabo/movie-review-data/
Table 6.16: Experiments were carried out while changing the magnitude values of the Galadriel system in five polynomial sequences normalised final Galadriel scores for each document. Then, the Galadriel cut-off values for three-class and four-class datasets were calculated for each experiment, and the movies classified reviews accordingly. Finally, the Galadriel results were compared with actual (gold standard) results and performance measures were calculated.

For three-class classification, tables 6.17 and 6.18 show (performance measures) precision, recall and f-score values for each class, with the values of magnitude nodes in arithmetic and polynomial sequences. Figures 6.30 and 6.31 show graphs of the performance measures against the different value of magnitude nodes in arithmetic and polynomial sequences. Table 6.31 shows sharp decrements in the graphs, when compared to table 6.30.

Similarly, tables 6.19 and 6.20 show performance measures for four-class classification. Figures 6.32 and 6.33 show the accompanying graphs.

<table>
<thead>
<tr>
<th>Ex</th>
<th>Nodes</th>
<th>Values of Magnitude Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>VERY-WK</td>
</tr>
<tr>
<td>1</td>
<td>$n^1$</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>$n^2$</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>$n^3$</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>$n^4$</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>$n^5$</td>
<td>1</td>
</tr>
</tbody>
</table>
### Table 6.17: Performance measures of three classes according to the values of the magnitude nodes in arithmetic sequences

<table>
<thead>
<tr>
<th>Nodes</th>
<th>scale-0</th>
<th>scale-1</th>
<th>scale-2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
</tr>
<tr>
<td>$n$</td>
<td>0.9124</td>
<td>0.7813</td>
<td>0.8418</td>
</tr>
<tr>
<td>$2n$</td>
<td>0.8571</td>
<td>0.7500</td>
<td>0.8000</td>
</tr>
<tr>
<td>$3n$</td>
<td>0.8387</td>
<td>0.6500</td>
<td>0.7324</td>
</tr>
<tr>
<td>$4n$</td>
<td>0.7778</td>
<td>0.5250</td>
<td>0.6269</td>
</tr>
<tr>
<td>$5n$</td>
<td>0.6923</td>
<td>0.4500</td>
<td>0.5455</td>
</tr>
</tbody>
</table>

### Table 6.18: Performance measures of three classes according to the values of the magnitude nodes in polynomial sequences

<table>
<thead>
<tr>
<th>Nodes</th>
<th>scale-0</th>
<th>scale-1</th>
<th>scale-2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
</tr>
<tr>
<td>$n^1$</td>
<td>0.9124</td>
<td>0.7813</td>
<td>0.8418</td>
</tr>
<tr>
<td>$n^2$</td>
<td>0.7073</td>
<td>0.7250</td>
<td>0.7160</td>
</tr>
<tr>
<td>$n^3$</td>
<td>0.6111</td>
<td>0.5500</td>
<td>0.5789</td>
</tr>
<tr>
<td>$n^4$</td>
<td>0.5000</td>
<td>0.3750</td>
<td>0.4286</td>
</tr>
<tr>
<td>$n^5$</td>
<td>0.4865</td>
<td>0.3600</td>
<td>0.4138</td>
</tr>
</tbody>
</table>
Figure 6.30: Graphs for performance measures of three classes of different magnitude values of nodes in arithmetic sequence
Figure 6.31: Graphs for performance measures of three classes and overall measures against different magnitude values of nodes in polynomial sequence
### Table 6.19: Performance measures of four classes according to the values of the magnitude nodes in arithmetic sequences

<table>
<thead>
<tr>
<th>Nodes</th>
<th>scale-0</th>
<th>scale-1</th>
<th>scale-2</th>
<th>scale-3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>$n$</td>
<td>0.7778</td>
<td>0.7000</td>
<td>0.7368</td>
<td>0.7143</td>
</tr>
<tr>
<td>$2n$</td>
<td>0.7308</td>
<td>0.6333</td>
<td>0.6786</td>
<td>0.5882</td>
</tr>
<tr>
<td>$3n$</td>
<td>0.7083</td>
<td>0.5667</td>
<td>0.6296</td>
<td>0.5405</td>
</tr>
<tr>
<td>$4n$</td>
<td>0.6522</td>
<td>0.5000</td>
<td>0.5660</td>
<td>0.4865</td>
</tr>
<tr>
<td>$5n$</td>
<td>0.6250</td>
<td>0.3226</td>
<td>0.4255</td>
<td>0.4250</td>
</tr>
</tbody>
</table>

### Table 6.20: Performance measures of four classes according to the values of the magnitude nodes in polynomial sequences

<table>
<thead>
<tr>
<th>Nodes</th>
<th>scale-0</th>
<th>scale-1</th>
<th>scale-2</th>
<th>scale-3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>$n^1$</td>
<td>0.7778</td>
<td>0.7000</td>
<td>0.7368</td>
<td>0.7143</td>
</tr>
<tr>
<td>$n^2$</td>
<td>0.6774</td>
<td>0.7000</td>
<td>0.6885</td>
<td>0.4231</td>
</tr>
<tr>
<td>$n^3$</td>
<td>0.5385</td>
<td>0.4667</td>
<td>0.5000</td>
<td>0.2424</td>
</tr>
<tr>
<td>$n^4$</td>
<td>0.4545</td>
<td>0.3333</td>
<td>0.3846</td>
<td>0.2000</td>
</tr>
<tr>
<td>$n^5$</td>
<td>0.5000</td>
<td>0.3000</td>
<td>0.3750</td>
<td>0.1579</td>
</tr>
</tbody>
</table>
Figure 6.32: Graphs for performance measures of the four classes and overall measures against different magnitude values of nodes in arithmetic sequence
Figure 6.33: Graphs for performance measures of the four classes and overall measures against different magnitude values of nodes in polynomial sequence
<table>
<thead>
<tr>
<th>$m$</th>
<th><strong>Arithmetic Sequences</strong> ($mn$)</th>
<th></th>
<th><strong>Polynomial Sequences</strong> ($n^m$)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
<td>F-Score</td>
<td>Precision</td>
</tr>
<tr>
<td>1</td>
<td>0.8286</td>
<td>0.8235</td>
<td>0.8260</td>
<td>0.8286</td>
</tr>
<tr>
<td>2</td>
<td>0.7600</td>
<td>0.7500</td>
<td>0.7549</td>
<td>0.6533</td>
</tr>
<tr>
<td>3</td>
<td>0.7300</td>
<td>0.7083</td>
<td>0.7190</td>
<td>0.5234</td>
</tr>
<tr>
<td>4</td>
<td>0.6847</td>
<td>0.6500</td>
<td>0.6669</td>
<td>0.4400</td>
</tr>
<tr>
<td>5</td>
<td>0.6314</td>
<td>0.5917</td>
<td>0.6109</td>
<td>0.4175</td>
</tr>
</tbody>
</table>

Table 6.21: Average performance measures of three classes according to the values of the magnitude nodes in both arithmetic and polynomial sequences.

Figure 6.34: Graphs of average performance measures of three classes

Finally, table 6.21 provides a comparison of average performance measures for three-class classification, when the values of Galadriel’s magnitude nodes change in arithmetic and polynomial sequences. Figure 6.34 shows the graphs for table 6.21. Similarly, table 6.22 and figure 6.35 show average performance measures and relevant graphs for three-class classification.

I also calculated mean absolute errors (MAE) for all experiments. Table ref{tab:MAEsum} shows MAE for each experiment separately and figure 6.36a presents the accompanying graph. The magnitude nodes in polynomial sequences produce larger error than in arithmetic sequences. Moreover, four-class classification indicates higher MAE than three-class classification.

Figure 6.36 shows the average f-score for both three- and four-class classification with magnitude nodes in both arithmetic and polynomial sequences separately. That is to say, Galadriel shows poor performance when the interval between magnitude nodes increases. Moreover, when the ratio of intervals between magnitude nodes increases, the system shows its worst performance. These changes in the intervals affect four-class classification more than three-class classification.
<table>
<thead>
<tr>
<th>$m$</th>
<th><strong>Arithmetic Sequences</strong> ($mn$)</th>
<th><strong>Polynomial Sequences</strong> ($n^m$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
</tr>
<tr>
<td>1</td>
<td>0.7419</td>
<td>0.7417</td>
</tr>
<tr>
<td>2</td>
<td>0.6556</td>
<td>0.6500</td>
</tr>
<tr>
<td>3</td>
<td>0.6292</td>
<td>0.6199</td>
</tr>
<tr>
<td>4</td>
<td>0.5860</td>
<td>0.5750</td>
</tr>
<tr>
<td>5</td>
<td>0.5423</td>
<td>0.5167</td>
</tr>
</tbody>
</table>

Table 6.22: Average performance measures of four-class classification according to the values of the magnitude nodes in both arithmetic and polynomial sequences.

(a) Value of magnitude nodes in arithmetic sequences

(b) Value of magnitude nodes in polynomial sequences

Figure 6.35: Graphs for average performance measures of four classes.

<table>
<thead>
<tr>
<th>$m$</th>
<th><strong>Arithmetic sequences</strong> ($mn$)</th>
<th><strong>Polynomial sequences</strong> ($n^m$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3-class</td>
<td>4-class</td>
</tr>
<tr>
<td>1</td>
<td>0.2167</td>
<td>0.3250</td>
</tr>
<tr>
<td>2</td>
<td>0.2583</td>
<td>0.4250</td>
</tr>
<tr>
<td>3</td>
<td>0.3250</td>
<td>0.4750</td>
</tr>
<tr>
<td>4</td>
<td>0.3917</td>
<td>0.5333</td>
</tr>
<tr>
<td>5</td>
<td>0.6167</td>
<td>0.6547</td>
</tr>
</tbody>
</table>

Table 6.23: Mean average error for both three- and four-class classification.
6.9 Summary of the Chapter

This chapter has discussed the integrated Galadriel system. I used the basic rules and algorithms from SO-CAL and OO to handle the valence shifters and merged them in an inheritance structured framework. The integrated Galadriel system consists of 7 models. Each model has rules and algorithms to handle different sentiment behaviour if lexical items and models are inherited from the other. The basic Galadriel system replicates the SO-CAL and OO systems separately, as discussed in chapter 5. However, the integrated Galadriel system is able to handle both SO-CAL and OO features and different level sentiment analysis tasks in the same framework.

Like any typical lexicon based approach, the integrated system uses a sentiment dictionary. However, it also tackles the irregular behaviour of lexical items based on their context. I used corpus based learning methodology to collect the sentiment behaviour and manually plugged it into the Galadriel sentiment dictionary, which I call the Galadriel base model. The integrated system gives more granularity. Finally, I carried out the evaluation of the integrated Galadriel with various domains across all sentiment analysis levels, such as sentence, document and aspect level tasks, and compared the performance of Galadriel with other sentiment analysis systems. I showed that the integrated Galadriel outperforms the baseline and produced better results.

Section 6.8 discussed the sensitivity and stability of Galadriel’s parametric feature, magnitude. I tested Galadriel’s performance while changing the magnitude values of the Galadriel lexicon. Different experiments were carried out while changing the values of the magnitude nodes, and Galadriel’s performance was tested. For sensitivity, I tested the performance, while slightly increasing and reducing all the magnitude nodes’ values equally. I showed that, for specific cut-off values (class
thresholds) Galadriel performs securely only when reducing the standard magnitude values of lexical items up to 0.25. For stability, I tested Galadriel’s performance while changing the intervals of magnitude nodes’ value. The experiments were carried out with the magnitude values of nodes in arithmetic and polynomial sequences. Finally, I showed that Galadriel shows relatively poor performance when the intervals are large.
Chapter 7

Summary and Conclusions

This thesis has explored an inheritance-based mechanism for sentiment analysis. First, I created a lexicon by structuring lexical items in an inheritance-based network based on sentiment behaviour. Then different sentiment models were created in an inheritance structure that can handle different types of lexical items and calculate sentiment score in an inheritance-based structure. A novel modelling framework was presented in chapter 4, which addressed the inheritance-based modelling methodology. In chapter 5, I also proposed a novel technique for the evaluation mechanism in regards to sentiment analysis. An integrated sentiment model was presented in chapter 6 and evaluation of the integrated system on different levels of sentiment analysis was provided. In this final chapter, I summarise the study and present the conclusion. Finally, I complete the chapter with a discussion of possible future work.

7.1 Summary

The DATR/ELF lexicon representation system can encode very complex information, including phonology, morphology, syntax and semantics. This research aimed to exploit this architecture for sentiment analysis. The starting point was an initial system called Galadriel 0.1, which is a simple sentiment framework using DATR/ELF. Using Galadriel, I set out to study two existing lexicon-based approaches to sentiment analysis and replicate their key analysis algorithms as ELF rules. I selected two previous lexicon-based methods, (Taboada et al., 2011)’s ‘Lexicon-Based Methods for Sentiment Analysis’ (the SO-CAL system) and (Ding et al., 2008)’s ‘A Holistic Lexicon-Based Approach to Opinion Mining’ (the OO system). I studied the algorithms and techniques which they used to build their systems and modelled them in Galadriel, using as far as possible the same datasets
and the features which were utilised in the existing systems. The additional SO-CAL features, weighting and multiple cut-offs, were not implemented in Galadriel, because these features are not appropriate for sentiment analysis. These features were implemented using external sources. However, my pre-evaluation process provides a similar effect provided by SO-CAL’s cut-off feature. The OO system also used external information to assign polarity for context-dependent words, whereas Galadriel uses only local information for that, as the current Galadriel version is not built to access external sources. The OO system is an aspect-based sentiment analysis system and it was built based on the assumption that the aspect terms are given. However, I used the Search Engine tool to extract the aspect terms using a training dataset.

I then evaluated Galadriel’s performance against each system. The evaluation shows the competitive results, which are overall f score of 0.83 for SO-CAL features and 0.90 for the OO features. So I have demonstrated the principle that sentiment knowledge can be modelled in the DATR/ELF inheritance framework. As a next step, I merged the techniques of SO-CAL and OO in Galadriel. From these analyses, an integrated inheritance model of sentiment knowledge of words was defined and extended to a model of sentiment analysis. In this way, the entire sentiment analysis task was coded as a ‘lexical description’ task. I also introduced insights from other approaches, in particular corpus-based learning techniques, into the model. To illustrate, I aimed to use Galadriel to handle phrases and irregular sentiment words such as aspect words that are commonly used in web documents, which can express an opinion, and sentiment idioms. I collected such phrases and words with examples derived from corpus data. I added them to the Galadriel lexicon as sentiment units. In this way, Galadriel’s inheritance-based lexicon has supported exceptions to general rules. Finally, the Galadriel lexicon was created in an inheritance-based structure based on the sentiment behaviour of lexical items. Then the different Galadriel models, with the identified techniques, were developed in an inheritance structure to calculate the overall semantic orientation of a text while considering valence shifters, such as negation and intensifiers. Galadriel’s inherited models help to reduce duplicate rules and algorithms handle linguistic features for sentiment calculation.

The complete developed Galadriel 1.0 system addresses different levels of sentiment analysis related to the current research area: document-level, sentence-level and aspect-level. The main properties of the Galadriel system involve calculating sentiment magnitude and polarity of text. The final Galadriel output for a document/text produces a real number (with a positive or negative sign), which is a regression model. I also introduced a calibration method that maps to the classes
as I discuss later in this section.

Previous researchers have handled context-dependent sentiment words in a variety of ways. I added a feature called context-polarity to identify context-dependent sentiment words in a text. This feature carries the text’s author’s sentiment polarity from the beginning to the end of the statement. Thus, the context-polarity feature expresses the author’s state of mind at every stage of the text, and the full scope of their sentiment. *Galadriel* assigns sentiment for context-dependent sentiment words according to the author’s actual views. The context-polarity feature explains the polarity of a lexical item within a sentence, regardless of its individual sentiment score; this provides a route for handling sarcasm and irony. Sentiment analysis tasks in comparative sentences are somewhat complicated. I used the grammatical particle *than*, which is used in comparison sentences, and modelled it with an added algorithm in *Galadriel*.

I defined a neutral class which can be differentiated from the mixed sentiment class. I demonstrated that I could model *Galadriel* to calculate positive and negative sentiment scores separately, and detect neutral sentiment text by introducing an extended *Galadriel* model.

In addition, I wanted to assess the *Galadriel* system against gold standard systems. In order to do that, I aimed to compare *Galadriel*’s outputs with gold standard systems’ outputs, and calculate precision, recall and f-score values. However, most of the gold standard systems return categorical data types, such as *positive*, *negative* and *neutral*, or 4/5-star scales. *Galadriel*, however, returns a numeric sentiment score. This is a real issue in the evaluation process. I came up with a new technique, able to calibrate *Galadriel*’s sentiment scores with the fixed, ordered classes that are returned by gold standard systems. I proposed a novel calibration method to set class thresholds to optimise performance by using a precision vs recall curve in chapter 5. I defined this calibration process as a pre-evaluation process. I also identified the parameters of *Galadriel* and tested its sensitivity and stability according to the class thresholds.

### 7.2 Discussion and Limitations

#### 7.2.1 Discussion

This discussion section presents some alternative methods which I could have employed when modelling *Galadriel*. In the aspect-level sentiment analysis task, I introduced the found-ASPECTi feature, and the values *true* or *fail* were assigned
by computing found right and found left ASPECT values. I could, however, have introduced a Galadriel feature called Aspect for each word’s lexical agent, with the value of the targeted aspect (e.g. SCREEN, PRICE, BATTERY, etc.). I could also have added a model to Galadriel that computes the aspect value of each lexical item in the sentence. Figure 7.1 shows Galadriel’s aspect features with their possible values.

Figure 7.1: Alternative method for calculating aspect score

For this project a base model in Galadriel was used. The base model operates at the word level, but ELF also provides also lex and phrase models which treat multi-word units as single entities, and this could be used to capture more advanced multi-word sentiment behaviour. When I model sentiment phrases in Galadriel, I model every single word of the phrase with the specific rules. A lex model(or lexeme model) contains lex phrases built out of base model tokens, and the next level is a phrase model which is built out of the lex model phrase. Instead of the base model, I could have used the phrase model to model sentiment phrases and idioms in Galadriel. The phrase model means the words in the phrase are subsumed by the first word of the phrase. For example, consider the phrase the bee’s knees in the following sentence:

‘Try this chocolate. It’s the bee’s knees, it really is.’

Figure 7.2: The polarity, magnitude and senti-score values of each word in the sentence

Figure 7.2 shows the calculation of polarity, magnitude and senti-score for each
word in the phrase. However, the current version of Galadriel is designed to calculate the total score of a text by adding the senti-score of a word to the total value of its previous word. In this way, the total value of the item next to knees (‘,’) is as follows:

\[ \text{tot} = \text{knees}_\text{tot} + \text{senti-score} \]
\[ = 0 + 0 \]
\[ = 0 \]

Therefore, calculation of the total would not give the expected value. However, this error is fixable. Moreover, the ELF has more advanced tools for identifying phrases, and it would be an interesting extension to Galadriel to use them.

Furthermore, a common critical question which arose in my study is ‘how reliable are the sentiment scores of the lexicon?’ or ‘why did you choose the particular sentiment dictionary?’ These are common questions for all lexicon-based approaches. The answer is, my study only aimed to build a sentiment analysis system using available sentiment lexicon dictionaries. I also could have used other sentiment lexicons, such as Sentiword, SocialSent, etc. Although their scoring range might be different to each other, their sentiment scales are similar. For example, consider the positive words good and excellent. The individual sentiment score of both words might be different in various sentiment dictionaries. However, the difference between the sentiment score of both words might be relatively equal. I aimed to calculate the sentiment scale of a text, and I calibrate the final score if it is necessary to compare my system with other systems. So my calibration technique does provide a proper answer to the questions. Although it is not possible to tell if the selected (dictionaries) scores are ‘right’ but I can work out how to map them onto classes, and hence compare them with both classification systems and other regression systems in a meaningful way.

7.2.2 Limitations

The project proposed a fine-grained sentiment analysis system using an inheritance-based lexicon approach. I built a sentiment framework, Galadriel, which uses the DATR/ELF inheritance mechanism. There are some limitations to this project, however.

As described in the discussion in chapter 6, Galadriel does not yield accurate results for some comparative sentences in document-/sentence-level analysis tasks. For example, the phrase ‘an iPhone is better than a Samsung phone’, if it appears in
a review of a Samsung phone (see section 6.7). This issue can be overcome by introducing a topic detection model in Galadriel, so that every single sentence will detect its subject.

Another obstacle of this project is processing text that contains misspelt sentiment words. The Galadriel models can only detect words and assign their features according to the Galadriel lexicon. Any words not in the Galadriel lexicon are assigned to the neutral (default) sentiment category. Therefore, any misspelt words are categorized as neutral. Informal tweets and sentences often contain either misspelt or shortened forms of text. However, I could add abbreviation and other standard shortened forms of words and phrases, or dynamic spelling correction, to the Galadriel lexicon. I also could use the ELF feature that incorporates support for spelling correction, providing both Soundex and Hunspell encodings for words, which could form the basis for supporting this.

I added a model to handle sentiment idioms, phrases and irregular words to the Galadriel system. I demonstrated that I populated the list of phrases and words from the various corpora, using corpus-based learning techniques. However, on this project, I did not attempt to accurately annotate the magnitude value for those phrases and words, as this is beyond the scope of this research. However, this could affect the final scores in Galadriel’s output.

### 7.3 Conclusion

The main goal of this thesis was to develop a sentiment analysis tool using an inheritance-based mechanism. I started with my research question:

*Can inheritance-based modelling techniques be used to improve the modelling of sentiment in a text?*

And I set the following objectives:

- **Objective 1**: To model sentiment knowledge using DATR’s inheritance mechanism by modelling existing lexicon-based approaches to sentiment analysis. Evaluate the effectiveness of the model and identify scope for improvement.

- **Objective 2**: To combine and extend models of existing systems to provide an innovative rule-based system using the inheritance-based model of sentiment knowledge.

- **Objective 3**: To refine the inheritance-based model by extending and/or overriding its rule-based system based on corpus analysis techniques.
• Objective 4: To evaluate the proposed model quantitatively in order to assess the effectiveness of inheritance-based modelling techniques for sentiment analysis.

In chapter 4, I discussed the modelling of sentiment knowledge using the inheritance mechanism and introduced a simple sentiment analysis framework, basic Galadriel, using the DATR/ELF inheritance-based mechanism. In chapter 5, I showed that the existing lexicon-based approaches can be modelled in Galadriel. I also proved that Galadriel yields similar or even better results by performing the evaluation (objective 1). Chapter 6 discussed the combination of both existing lexical approaches in Galadriel (objective 2). I also extended the combined model by adding more rules that also can handle irregular lexical items and produced an integrated Galadriel system (objective 3). Finally, chapter 6 discussed the production of Galadriel 1.0, evaluated qualitatively against the different levels of sentiment analysis. The final results provided that f-score of 0.8284 on sentence-level, 0.78 (three class)/0.75 (four class) on document level and 0.8079(Restaurant)/0.7464(Laptop) on aspect-level. This shows that Galadriel outperforms the base-line systems (objective 4).

The thesis also contributes to evaluation mechanisms in the field by proposing a novel calibration method for the evaluation of sentiment analysis (chapter 5). Traditional sentiment analysis systems represent fixed sentiment classes for a given piece of text. This creates a challenge for comparing such systems with Galadriel; in particular assessing Galadriel’s numerical scores against datasets that use fixed classes is difficult because the numerical outputs have to be mapped on to the ordered classes. Hence, I proposed a novel calibration technique that uses precision vs recall curves to set class thresholds to optimize Galadriel’s performance against the gold standard systems.

7.4 Future Work

This thesis has demonstrated that inheritance-based modelling techniques can be used to improve the modelling of sentiment in a text by building a sentiment analysis system, Galadriel. I have showed that Galadriel produces comparable results to other systems. However, there are many opportunities for the research in this thesis to be extended. This section presents some of those possibilities.

The most immediate consideration would be adding a model for handling figurative language such as sarcasm and irony to the Galadriel system. As I mentioned in the summary, the sentiment scope of each word of a text is identified by Galadriel’s context-polarity feature. The current work identified the sentiment of the neutral
words in a context (context-dependent sentiment words). The new model could be added to reassign/recalculate the sentiment score of a polar word according to its sentiment scope. Moreover, some figurative language (e.g. similes) is used to show the intensity of words (e.g. ‘as hot as an oven’, ‘as big as a mountain’, ‘as fast as the wind’). \textit{Galadriel} could handle such similes using the \textit{Galadriel} magnitude and factor (intensifier) values.

The issues of handling anaphora and cataphora might reflect mainly on aspect-level sentiment analysis, because co-references may be used to refer to an aspect which has been mentioned in its antecedent expression. Although I did not include any techniques in this respect, some co-references could be easily modelled in \textit{Galadriel}. This would a useful technique to introduce to the next version of \textit{Galadriel}.

Another interesting direction would be building a recommender system on top of the \textit{Galadriel} system. Aspect-level sentiment analysis tasks in \textit{Galadriel} extract the sentiment of the author towards each aspect (attribute) of a product in a product review. I could add an extra model or merge \textit{Galadriel} with a system that has a database of products and their attributes. Then, according to the \textit{Galadriel} output (sentiment class), the new model would suggest products about which the author is likely to have a positive sentiment.
Appendix A

The *Galadriel* Code
% simple sentiment handling

#vars $model:
  base params lex phrase % built-in models
  sent1 sent2 sent3 sent4 sent5 sent6 sent7 % additional models
.
% load DATR library Eval (for doing simple maths)
#uses Eval.
% galadriel.sentiment.WORD-LOOKUP:<$word $index>
% lookup node name for words which do not have their own definitions
% galadriel.sentiment.WORD-LOOKUP:
< > == 'galadriel.sentiment.ROOT'
% galadriel.sentiment.ROOT
%
% root node for all words in galadriel
% default score is zero
% default total adds the score here to the total from the previous word
galadriel.sentiment.ROOT:
< > == galadriel.LEXROOT
% extensions to base model
<base type> == neutral
<base pol> == 0
<base mag> == 0
<base factor> == 1
<base senti-score> == 0
<base context-pol> == 0
<neg-context> == no
<base block-context> == no
<base ques-context> == no
<base total> == 0
%
% Only for aspec-based sentiment analysis tasks
<sent2 total-screen> == <here $model total screen>
<sent2 total-battery> == <here $model total battery>
<sent2 total-speaker> == <here $model total speaker>
<sent2 found-screen> == <here $model found screen>
<sent2 found-battery> == <here $model found battery>
<sent2 found-speaker> == <here $model found speaker>

<sent> == <here base>
<sent1 senti-score> == Eval:< <here sent1 mag> * <here sent1 pol> .>
<sent1 total> == Eval:< <here sent1 senti-score> + <here sent1 prev sent1 total> .>

<sent1 neg-context> == < IFEQ:< <here sent1 type .> negation THEN case negation-THEN case neg-found ELSE case negation-context .> >
<case neg-found> == no
<case neg-context> == yes
<case neg-context> == < IFEQ:< <here sent1 prev sent1 type .> boundary THEN case skip-found ELSE test mark-context .> >
<test mark-context> == <here sent1 prev sent1 mark-context>

<sent1 context-pol> == < IFEQ:< <here sent1 type .> positive THEN test neg-con1 ELSE case nega .> >
<test neg-con1> == < IFEQ:< <here sent1 neg-context .> yes THEN case negation2 ELSE case default2 .> >
<case negation2> == Eval:< <here sent1 context-pol> * -1 .>
<case default2> == <sent1 context-pol>
<sent2 pol> == < IFEQ:< <here sent1 type .> context THEN case con-found ELSE case non-context .> >
<case con-found> == < IFEQ:< <here sent1 neg-context .> yes THEN case negation2 ELSE case default2 .> >
<case negation2> == Eval:< <here sent1 context-pol> * -1 .>
<case default2> == <sent2 context-pol>

<sent3 senti-score> == Eval:< <here sent2 senti-score> + <here sent2 prev sent2 total> .>
<sent3 total> == Eval:< <here sent3 senti-score> + <here sent3 prev sent3 total> .>

<sent3 neg-context> == < IFEQ:< <here sent3 type .> >
<case neg-found> == no
<case neg-context> == yes
<case neg-context> == < IFEQ:< <here sent3 prev sent3 type .> boundary THEN case skip-found ELSE test neg-context .> >
<test neg-context> == <here sent3 prev sent3 neg-context>
+ <here sent2 mag>  .>
<case default21> == <here sent2 mag>

% sent4 - negation rules applies in this model- model that looks for negation context and change the polarity and magnitude
<sent4> == <here sent3>
<sent4 senti-score> == Eval:< <here sent4 mag> * <here sent4 pol> .>  
<sent4 total> == Eval:< <here sent4 senti-score> + <here sent4 prev sent4 total> .>  
<sent4 pol> == < IFEQ:< <here sent3 neg-context .> yes THEN case negation-pol ELSE case default-pol4 .> >  
<case default-pol4> == <here sent4 pol>
<case negation-pol> == Eval:< <here sent3 pol> * -1 .>  
<sent4 mag> == < IFEQ:< <here sent3 neg-context .> yes THEN case negation-mag ELSE case default-mag4 .> >  
<case default-mag4> == <here sent3 mag>
<case negation-mag> == Eval:< <here sent3 mag> * 0.3 .>  

% sent5 - model for blocking words- model that looks for block context words and change the polarity
<sent5> == <here sent4>
<sent5 senti-score> == Eval:< <here sent5 mag> * <here sent5 pol> .>  
<sent5 total> == Eval:< <here sent5 senti-score> + <here sent5 prev sent5 total> .>  
<sent5 pol> == < IFEQ:< <here sent4 block-context .> yes THEN case block-pol ELSE case default5 .> >  
<case default5> == <here sent4 pol>
<case block-pol> == 0  

% sent6 - model for interrogative sentence- model that looks for question context and change the polarity
<sent6> == <here sent5>
<sent6 senti-score> == Eval:< <here sent6 mag> * <here sent6 pol> .>  
<sent6 total> == Eval:< <here sent6 senti-score> + <here sent6 prev sent6 total> .>  
<sent6 ques-context> == < IFEQ:< <here sent6 type.> skip THEN case-default6 ELSE case-false .> >  
<case-default6> == **true**
<case-false> == **false**  

% sent7 - model for identify aspects and calculates score towards the each aspects
<sent7> == <here sent6>
<sent7 senti-score> == Eval:< <here sent7 mag> * <here sent7 pol> .>  
<sent7 total> == Eval:< <here sent7 senti-score> + <here sent7 prev sent7 total> .>  

% galadriel.sentiment.POSITIVE
% root node for positive sentiment words
galadriel.sentiment.POSITIVE:  
\( <> == \) galadriel.sentiment.ROOT:<>  
<base type> == positive  
<base pol> == 1  
<base mag> == 0  

% galadriel.sentiment.NEGATIVE
% root node for negative sentiment words
galadriel.sentiment.NEGATIVE:  
\( <> == \) galadriel.sentiment.ROOT:<>  
<base type> == negative  
<base pol> == -1  
<base mag> == 0  

% galadriel.sentiment.NEUTRAL
% root node for neutral sentiment words
galadriel.sentiment.NEUTRAL:  

% sent7 found-right $feature> == < IF EQ:< <here sent7 word.> $feature THEN feature-found ELSE test-skip .> .>  
<feature-found $feature> == **true**  
<test-skip $feature> == < IF EQ:< <here sent7 prev type.> skip THEN feature-fall ELSE next-feature .> .>  
<feature-fall $feature> == **false**  

% sent7 found-left $feature> == < IF EQ:< <here sent7 word.> $feature THEN feature-found ELSE test1-skip .> .>  
<test1-skip $feature> == < IF EQ:< <here sent7 prev type.> skip THEN feature-fail ELSE pre-feature .> .>  
<feature-fail $feature> == **false**  
<pre-feature $feature> == < here sent7 prev sent7 found-right $feature>  
<sent7 found $feature> == < IF:< OR:< <sent7 found-left $feature .> .> THEN case-default ELSE case-false .> .>  
<case-default> == **true**  
<case-false> == **false**  

% sent7 total $feature> == Eval:< <here sent7 senti-score $feature> + <here sent7 prev sent7 total $feature> .>  

% sent7 total $feature> == Eval:< <here sent7 senti-score $feature> + <here sent7 prev sent7 total $feature> .>  

% sent7 found-right $feature> == < IF EQ:< <here sent7 word .> $feature THEN feature-found ELSE test-skip .> .>  
<feature-found $feature> == **true**  
<test-skip $feature> == < IF EQ:< <here sent7 prev type .> skip THEN feature-fall ELSE next-feature .> .>  
<feature-fall $feature> == **false**  

% sent7 found-left $feature> == < IF EQ:< <here sent7 word .> $feature THEN feature-found ELSE test1-skip .> .>  
<test1-skip $feature> == < IF EQ:< <here sent7 prev type .> skip THEN feature-fail ELSE pre-feature .> .>  
<feature-fail $feature> == **false**  
<pre-feature $feature> == < here sent7 prev sent7 found-right $feature>  
<sent7 found $feature> == < IF:< OR:< <sent7 found-left $feature .> .> THEN case-default ELSE case-false .> .>  
<case-default> == **true**  
<case-false> == **false**  

% sent7 total $feature> == Eval:< <here sent7 senti-score $feature> + <here sent7 prev sent7 total $feature> .>  

% galadriel.sentiment.POSITIVE
% root node for positive sentiment words
galadriel.sentiment.POSITIVE:  
\( <> == \) galadriel.sentiment.ROOT:<>  
<base type> == positive  
<base pol> == 1  
<base mag> == 0  

% galadriel.sentiment.NEGATIVE
% root node for negative sentiment words
galadriel.sentiment.NEGATIVE:  
\( <> == \) galadriel.sentiment.ROOT:<>  
<base type> == negative  
<base pol> == -1  
<base mag> == 0  

% galadriel.sentiment.NEUTRAL
% root node for neutral sentiment words
galadriel.sentiment.NEUTRAL:
<> == galadriel.sentiment.ROOT:<>
<base type> == neutral
<base pol> == 0
<base mag> == 0
.
% galadriel.sentiment.CONTEXT
% root node for context dependent sentiment words
galadriel.sentiment.CONTEXT:
<> == galadriel.sentiment.NEUTRAL:<>
<base type> == context
.
% galadriel.sentiment.INTENSIFIER
% root node for intensifier words
galadriel.sentiment.INTENSIFIER:
<> == galadriel.sentiment.NEUTRAL:<>
<base type> == intensifier
.
% galadriel.sentiment.NEGATION
% root node for negative sentiment words
galadriel.sentiment.NEGATION:
<> == galadriel.sentiment.NEUTRAL:<>
<base type> == negation
.
% galadriel.sentiment.MARK
% root node for negative sentiment words
galadriel.sentiment.MARK:
<> == galadriel.sentiment.NEUTRAL:<>
<base type> == mark
.
% galadriel.sentiment.BOUNDARY
% root node for negative sentiment words
galadriel.sentiment.BOUNDARY:
<> == galadriel.sentiment.NEUTRAL:<>
<base type> == boundary
.
% galadriel.sentiment.QUESTION
% root node for negative sentiment words
galadriel.sentiment.QUESTION:
<> == galadriel.sentiment.NEUTRAL:<>
<base type> == question
.
% galadriel.sentiment.FEATURE
% root node for negative sentiment words
galadriel.sentiment.FEATURE:
<> == galadriel.sentiment.NEUTRAL:<>
<base type> == feature
.

-5-

C:\Users\rpe\Desktop\Galadriel-code\Galadriel-code\full.txt
13 September 2017 21:17
13 September 2017 21:17
C:\Users\rpe\Desktop\Galadriel-code\Galadriel-code\full.txt
13 September 2017 21:17

-6-
References


Alexandra Balahur, Ralf Steinberger, Mijail Kabadjov, Vanni Zavarella, Erik Van Der Goot, Matina Halkia, Bruno Pouliquen, and Jenya Belyaeva. Sentiment


Samuel Brody and Noemie Elhadad. An unsupervised aspect-sentiment model for online reviews. In Human Language Technologies: The 2010 Annual Conference


Govin Gaikwad and Deepali J Joshi. Multiclass mood classification on twitter using lexicon dictionary and machine learning algorithms. In *Inventive Computation*


Matthew L. Jockers. Revealing sentiment and plot arcs with the syuzhet package, February 2015. URL [http://www.matthewjockers.net/2015/02/02/syuzhet/](http://www.matthewjockers.net/2015/02/02/syuzhet/).


Matthias Landt. Sentiment analysis as a tool for understanding fiction. 2010.

Peter C. R. Lane, Daoud Clarke, and Paul Hender. On developing robust models for favourability analysis: Model choice, feature sets and imbalanced data. *Decis.*


RR Mavljutov and NA Ostapuk. Using basic syntactic relations for sentiment analysis, 2013.


Harvey J Miller and Jiawei Han. *Geographic data mining and knowledge discovery*. CRC Press, 2009.


Phan Trong Ngoc and Myungsik Yoo. The lexicon-based sentiment analysis for fan page ranking in facebook. In The International Conference on Information


Georgios Paltoglou and Mike Thelwall. More than bag-of-words: Sentence-based

Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjec-
tivity summarization based on minimum cuts. In *Proceedings of the 42nd annual

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment
categorization with respect to rating scales. In *Proceedings of the 43rd Annual
Meeting on Association for Computational Linguistics*, ACL ’05, pages 115–124,
3115/1219840.1219855. URL http://dx.doi.org/10.3115/1219840.1219855.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment
classification using machine learning techniques. In *Proceedings of the ACL-02
conference on Empirical methods in natural language processing-Volume 10*, pages

Hitesh Parmar, Sanjay Bhanderi, and Glory Shah. Sentiment mining of movie
reviews using random forest with tuned hyperparameters. 2014.

James W Pennebaker, Roger J Booth, and Martha E Francis. Linguistic inquiry

Livia Polanyi and Annie Zaenen. Contextual valence shifters. In *Computing attitude

Carl Pollard and Ivan A Sag. *Head-driven phrase structure grammar*. University of

C. Pong-Inwong and K. Kaewmak. Improved sentiment analysis for teaching eval-
2nd IEEE International Conference on Computer and Communications (ICCC)*,

Jay M Ponte and W Bruce Croft. A language modeling approach to information
retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh
Manandhar, Mohammed AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing
Qin, Orphée De Clercq, et al. Semeval-2016 task 5: Aspect based sentiment
Association for Computational Linguistics, 2016.


Hassan Saif, Miriam Fernandez, Yulan He, and Harith Alani. Evaluation datasets for twitter sentiment analysis: a survey and a new dataset, the sts-gold. 2013.


Velislava Stoykova. Representing lexical knowledge for bulgarian inflectional morphology in datr. 2010.


Yves Vanrompay, Mario Cataldi, Marine Le Glouanec, and Myriam Lamolle. Sentiment analysis for dynamic user preference inference in spoken dialogue systems. 2014.


Ian H Witten, Craig G Nevill-Manning, and Sally Jo Cunningham. Digital libraries based on full-text retrieval. ERIC, 1996.


JJ Zhai, Nicholas Cohen, and Anand Atreya. Sentiment analysis of news articles for financial signal prediction, 2011.


