



Anglia Ruskin
University

Entity Aspect Extraction Using Double Propagation

SID:1302575

MOD002791 Final Project

Final Project Report

BEng (Hons) Computer Science

Submitted: April 2017

Abstract

Automated sentiment analysis has in the last years become an important tool for companies to keep track on how the public has perceived their products. Aspect level sentiment analysis provides the most detailed tools to analyse which parts of the product or service opinions have been given about. An important part of this is extracting from the text the entity aspects which opinions have been given about. Multiple tools for this have been proposed, but all current methods are limited and imperfect. Syntax based algorithm Double Propagation is a promising tool for the task, but is semi-supervised originally requiring a seed set of adjectives to start the double propagation process. This project aims to change that by implementing a method that creates the seed set from the analysed data automatically.

Table of Contents

Abstract.....	ii
Table of Contents.....	iii
List of Figures.....	v
List of Tables	vi
1.0 Introduction.....	8
1.1 Aims of the Study	8
1.2 Background.....	9
1.2.1 Aspect Level Sentiment Analysis.....	10
1.2.2 Two main tasks of aspect level analysis.....	11
2.0 Literature Review	12
2.1 Supervised Methods.....	12
2.2 Unsupervised Methods.....	13
2.2.1 Frequency based methods.....	13
2.2.2 Syntax based methods.....	14
2.3 Summary.....	16
3.0 Methodology.....	17
3.1 Used Software.....	17
3.1.1 Python.....	17
3.2.1 spaCy.....	17
3.2 Agile Methodology.....	18
3.3 System Restrictions.....	19
3.4 Data.....	20
3.5 System Design.....	20
3.6 Summary.....	26
4.0 Implementation.....	27
4.1 Pre-processing.....	27

4.2 Adjective Seed Set Creation.....	27
4.2.1 Parsing.....	28
4.2.2 Frequency Distribution.....	30
4.3 Double Propagation Algorithm.....	31
4.3.1 Algorithm Overview.....	31
4.3.2 Dependency Tags.....	34
4.3.3 Extraction Rules.....	35
4.3.4 Dependency Parsing.....	37
4.3.5 Extraction Rule Methods – Design and Implementation.....	39
4.4 Aspect Pruning.....	44
4.5 Summary.....	45
5.0 Testing.....	46
5.1 Functionality Testing.....	47
5.2 Accuracy Testing.....	48
5.3 Summary.....	52
6.0 Discussion.....	53
6.1 Accuracy and precision of the algorithm.....	53
6.2 Double propagation and different text domains.....	55
6.3 Algorithm Replication Issues.....	56
6.4 Summary.....	57
7.0 Conclusions.....	58
References	lix
Appendix.....	lxii

List of Figures

Figure 3.1: System Diagram.....	21
Figure 3.2: Use Case Diagram.....	22
Figure 3.3: Class Diagram.....	24
Figure 3.4: Aspect Extraction Sequence Diagram.....	25
Figure 4.1: Double Propagation Activity Diagram.....	33
Figure 4.3: Extraction Rule 1.2.....	40
Figure 5.1: Summary of accuracy testing results.....	52
Figure 6.1: Accuracy and Precision by Dataset.....	54
Figure 6.2 : Mean Accuracy and Precision.....	54

List of Tables

Table 3.1 Use Case Description.....	23
Table 4.1: Dependency Tags.....	35
Table 4.2: Extraction Rules	36
Table 4.3: Dependency Parsing Example.....	38
Table 5.1: Functionality results testing of the double propagation algorithm.....	47
Table 5.2: Dataset 1 Apple iPod.....	48
Table 5.3: Dataset 2 Canon100.....	49
Table 5.4: Dataset 3 Apex DVD player.....	50
Table 5.5 Dataset 4 Nokia mobile phone.....	51

1.0 Introduction

The rise of internet brought with itself a large new domain for people to express their opinions about products, services and even politicians online.

This opinionated data and the opinions contained within it is an important way for companies and organizations to learn how their service or product is perceived by the public, and for the public to learn how other customers have liked the product.

The amount of this kind of data is increasing daily to the point where analysing it by hand would be an impossible task for a human to do, so automated sentiment analysis tools have been rapidly developed.

Aspect level sentiment analysis tools provide the most detailed information on what parts of the product opinions are given about. Aspect level sentiment analysis combines multiple tasks, one of the major ones being Entity Aspect Extraction. Multiple solutions for entity aspect extraction have been previously proposed. This report takes a semi-supervised entity aspect extraction method, and develops it into an unsupervised method.

1.1 Aims of the study

The aim of this work is to implement semi-supervised entity aspect extraction algorithm Double Propagation as first implemented by Qiu, Liu, Bu and Chin in their 2011 paper *“Opinion word Expansion and target extraction through double propagation”* (Qiu et al, 2011) in a way that makes it unsupervised by implementing a seed set creation method as a part of the algorithm. This will diminish the need to provide a seed set of words to start off the double propagation process.

Further aims of this work are:

- To research existing entity aspect extraction methods for aspect level sentiment analysis.
- Compare unsupervised and supervised aspect extraction methods and the results achieved with them.
- Test the implemented algorithm with and without the seed set creation method and analyse the findings to determine whether it affects the performance of the algorithm.

1.2 Background

Sentiment analysis, sometimes referred to as Opinion Mining, refers to automated analysis of people's opinions and emotions towards entities such as products or companies, usually from digital data such as product reviews or social media texts. (Liu, 2012). It has been widely studied and implemented in three levels: Document level, where it is assumed that a text document, such as a product review, contains single positive or negative sentiment towards a single entity. (Feldman, 2013; Lin, He, 2009; Bhatla et al, 2015). Sentence level, where a sentence or a phrase within a sentence is assumed to contain positive or negative sentiment towards a single entity.(Feldman, 2013; Tracstrom, McDonald,2011; Wilson, Wiebe, Hoffman, 2005) and Aspect level, where it is assumed that an opinions are given not only about an entity as whole, but also about aspects of an entity (for example screen of a phone would be an aspect of entity phone), and aims to find out which aspects positive or negative opinions have been given about.(Liu, 2012; Schouten, Fransincar,2015).For industry applications aspect level analysis is seen as the most useful, as it gives the most detailed information on which parts of product or a service are liked or disliked by the customers.

1.2.1 Aspect Level Sentiment Analysis

Aspect level sentiment analysis aims to find from an opinionated document(s) which aspects, or parts of an entity opinions have been given about. For example, a review of a smartphone could consist of opinions about the operating system, screen and battery life of the phone. The analysis is based on the idea that an opinion consists of two key elements: An opinion and its target. For example, in a sentence “I think this actor is brilliant” opinion would be brilliant, and target the actor. Or a sentence “I like this phone, but I think the screen is way too big”, which contains two opinions; one positive one (like, phone) and one negative one (screen (of the phone), too big). Opinion targets are either an entity (The actor in the example above), or sub-parts of an entity, its aspects (the screen in the example above). So, following this, the aim of successful sentiment analysis tool is to discover from a document all entities and aspects of entities which opinions have been given about, and the sentiment polarity of the opinions towards them. (Liu, 2012).

1.2.2 Two main tasks of aspect level analysis

Aspect level analysis has two main tasks: Entity aspect extraction and aspect sentiment classification. (Schouten, Fransincar, 2015). Entity aspect extraction aims to discover all aspects of an entity from the text, and aspect sentiment classification aims to discover the polarity (positive, negative, neutral) of the opinion given about the aspects. Although the model implemented for this project does simultaneous entity aspect extraction and sentiment word extraction, this project concentrates on entity aspect extraction, leaving aspect sentiment classification to future improvements. Different methods of entity aspect extraction are discussed in detail below.

Chapter 2 of this report will be a literature review comparing currently available aspect extraction methods. In chapter 3 design of the sentiment analysis system utilizing the algorithm selected in chapter 2 will be described. Chapter 4 will show the implementation of the system. Chapter 5 contains testing results of functionality and accuracy testing performed. The testing results will be then discussed in chapter 6.

2.0 Literature review

In this chapter, different aspect extraction algorithms will be discussed and compared to find the algorithm most suitable for this system. The advantages and disadvantages of supervised and unsupervised algorithms will be discussed and the testing results achieved with different algorithms will be discussed.

Multiple methods for aspect level sentiment analysis and solving both tasks as described above have been proposed. These methods can roughly be divided into two categories: Supervised and Unsupervised methods.

Supervised methods refer to a group of statistical and information classifier methods which use trained classifiers and pre-tagged training data. Unsupervised methods don't use trained classifiers. (Chaovalit, Zhou, 2005).

2.1 Supervised Methods

Different statistical and information classifier methods have been proposed for aspect and sentiment word extraction. All supervised methods use pre-tagged training data to train the model.

Conditional Random Fields, Naïve Bayes classifiers, Hidden Markov Models and Support Vector Machines have all been used for the task. Jin and Ho used lexicalized Hidden Markov Models (HMM), combining linguistically information such as Part-Of-Speech information with original HMM's, mining both explicit and implicit aspects and opinion words. (Jin, Ho, 2009). Jacob and Gurevych used Conditional Random Fields for their application. They used training data from multiple domains to tackle to problem of domain portability. (Jacob, Gurevych, 2010). Yu, Zha and Chua used One-Class Support Vector Machines together with sentence parsing information for aspect

extraction. Their model also included an aspect ranking system to remove false results. (Yu, Zha, Chua, 2011).

While good results have been achieved with supervised methods, they do always need manually annotated training data, creation of which can be a large task, and which must be done again every time the classifier is used on a different domain. Thus, the initial implementation and set up requires more manhours than unsupervised methods. Also, due to being trained with data from a certain domain, supervised methods are not easily transferrable to other kinds of domains, for example using a system trained with twitter data to analyse news feeds.

2.2 Unsupervised Methods

Unsupervised methods work without training data, making them easier to implement and easier to move from domain to domain.

Unsupervised methods can be roughly divided into two categories: Frequency based methods and Syntax based methods. (Schouten, Frasincar, 2015)

2.2.1 Frequency Based Methods

Frequency based methods work by the observation that a limited set of words occurs in reviews of a specific product more often than the rest of the vocabulary. For example, the word ‘actor’ would occur more often in a set of movie reviews than generally in the English language. (Hu. Liu, 2004). By exploiting this and finding the nouns (with the assumption that entity aspects are often nouns, other words such as adverbs can be aspects as well, and some methods do include them in the process) which occur in the analyzed text more often than they statistically should, a set of aspects can be found. Hu and Liu used this method in their article *Mining and Summarizing Customer Reviews*

(Hu, Liu, 2004). Scaffidi et al based their sentiment analysis tool Red Opal on Noun Frequency aspect extraction as well. (Scaffidi et al, 2007).

In a survey to done by Schouten and Frasincar, Liu's frequency based model achieved 72% precision, whereas Scaffidi et al's model achieved between 85% and 90% precision. (Schouten, Frasincar, 2015). However, both models were not tested with the same dataset, so the survey is only approximate. Frequency based methods require separate opinion word extraction methods to be implemented, as this method only extracts entity aspects from the text. Also Frequency based methods tend to only concentrate on finding the most frequent aspects in the set, thus missing some of the infrequent aspects.

2.2.2 Syntax Based Methods

Syntax based methods find aspects based on the syntactic relationships between words in a sentence, especially the dependency relationships between opinions (often appearing as adjectives) their targets (often appearing as nouns or noun phrases). In English language opinion – target relationships can be described as a limited set of possible dependency relationships between nouns and adjectives, and aspects can thus be found by extracting the opinion targets following these relationships. For example, in a sentence “*This phone has a great screen*” when it is known that the main entity is phone, screen can be extracted following that the word has between them implies the screen being part of the entity phone. (Tellaamudhan, Suresh, Raghavi, 2016).

Double propagation is a syntax based method, which falls in the category of unsupervised methods, although some implementations are semi-supervised requiring initial set of seed words to start the extraction process.

This method extracts aspects and opinion words simultaneously by exploiting the relationships between opinion words and their targets, as described above. It is

assumed, that adjectives are being used to imply emotion towards entity aspects. If we further assume that aspects words are often nouns, we can extract aspects by finding nouns which are in a certain linguistic relationship with an adjective, and vice versa. (Qiu et al, 2009). Qiu et al first implemented Double Propagation model in 2009. They used a seed set of adjectives, which were used as a start point for the double propagation process. First they extracted the nouns which were dependent on the adjectives in the seed set, and in following iterations over the text extracted new adjectives using the extracted nouns, and new nouns using the extracted new adjectives. Their method worked well, but falls in the category of semi-supervised, requiring the initial opinion word set for the beginning. Zhang et al improved the algorithm by adding additional language rules to be used for the extraction process, as well as an aspect ranking system to remove false positives from extracted features. (Zhang et al, 2010). The new feature ranking addition ranks out more rarely mentioned aspects, which although not discussed as often as other ones, are still important. Qiu et al's algorithm achieved 88% precision in the Schouten and Frasincar survey, where Zhang's algorithm achieved 78%. (Schouten, Frasincar, 2015). Tests were not done using the same dataset. Qiu et al's original double propagation algorithm was chosen for this project. Due to the limitations of supervised methods as discussed above, and the difficulty of transferring them from domain to domain, it was decided that unsupervised methods would be used. From unsupervised methods, the fact that double propagation does aspect extraction and sentiment word extraction simultaneously made it the best candidate for the project. This method also performs as well or better than other unsupervised methods. (Schouten, Frasincar, 2015).

2.3 Summary

In this chapter, different aspect extraction algorithms were discussed. It was found out, that although supervised methods do provide good results, the requirement of training data makes them more difficult to implement and harder to transfer from domain to domain. It was decided, that an unsupervised method would be used. From available unsupervised algorithms, syntax based double propagation algorithm was chosen for implementation. Good results ranging from 78% to 88% precision have been achieved with the algorithm, and it achieves two main tasks of aspect level analysis, aspect extraction and emotion word extraction simultaneously.

3.0 Methodology

Previous chapter showed the double propagation algorithm to be best for this system.

This chapter details the design of a sentiment analysis software implementing the algorithm. The used software and methodology will be detailed, and the design of each element of the system will be discussed in detail.

3.1 Used Software

3.1.1 Python

The system was implemented using 64-bit Python 2.7.13 together with spaCY dependency parser.

64-bit version of the language was chosen, as spaCY is only available for this version of Python.

Python as a programming language was selected because it has excellent functionality for processing linguistic data (Bird, Klein, Loper, 2009). Powerful dependency parser and other Natural Language Processing (NLP) tools are also available for free for Python. Programming was done using Python GUI.

3.1.2 spaCy

As the algorithm depends on dependency relationships between words in the text, a dependency parser was required. spaCy is the most effective and the fastest one currently available for Python for free, and it does not require an API to work with this language like other popular dependency parsers such as Stanford Parser do. SpaCy also does other required NLP operations such as Point-Of-Speech (POS) tagging, sentence splitting, tokenization and dependency parsing simultaneously, making it very effective for this project.

3.2 Agile Methodology

The system proposed for this project consists of 5 independent “modules” which were developed independently from one another using the agile methodology. Although the team aspects of Agile could not be implemented, the Iterative, incremental and evolutionary aspects of it were used throughout. Software development was done in spurts each lasting from few days to few weeks, consisting the design, development and acceptance testing of one “module” of the system. Each spurt ended with a functional piece of software. Agile was chosen because of the adaptive nature of this methodology. During each spurt, new information and ideas were discovered, and with agile methodology it was easy to take them into consideration with next spurts and the design of the other modules. This picture shows how the agile methodology works in several short spurts of planning, designing, building and testing a part of the whole software.



(Srivastava, 2017) ¹

¹ LinkedIn, 2017. *What is Agile Methodology? Distantvantage of the waterfall method.* [Online] Available at: <https://www.linkedin.com/pulse/what-agile-methodologydisadvantage-waterfall-model-bikesh-srivastava> [Accessed 22 April 2017]

In contrast to another popular software development methodology the waterfall method.



(Waterfall software development, 2014)²

Where the whole system is first designed, and then developed per the initial design.

3.3 System Restrictions

The proposed system extracts entity aspects and opinion words from a set of product reviews. This is done with certain limitations:

It is assumed that entity aspects are single nouns and opinion words are single adjectives. Only explicit aspects are considered. This means aspects explicitly named in the text. For example, both sentences: “This phone fits into a pocket nicely.” And “I like the size of

² Waterfall Software Development, 2014. *Waterfall software development*. [Online] Available at: <http://waterwaterfall.blogspot.co.uk/2014/12/waterfall-software-development.html> [Accessed 22 April 2017]

this phone” both discuss the aspect size of a phone, but only the second sentence is in the scope of this project.

Only entity aspects opinions are given about are considered. This means that only aspects which are connected to opinion words are considered. Other entity aspects may be present on the text, but if no opinion is given about them, they are out of the scope of this project.

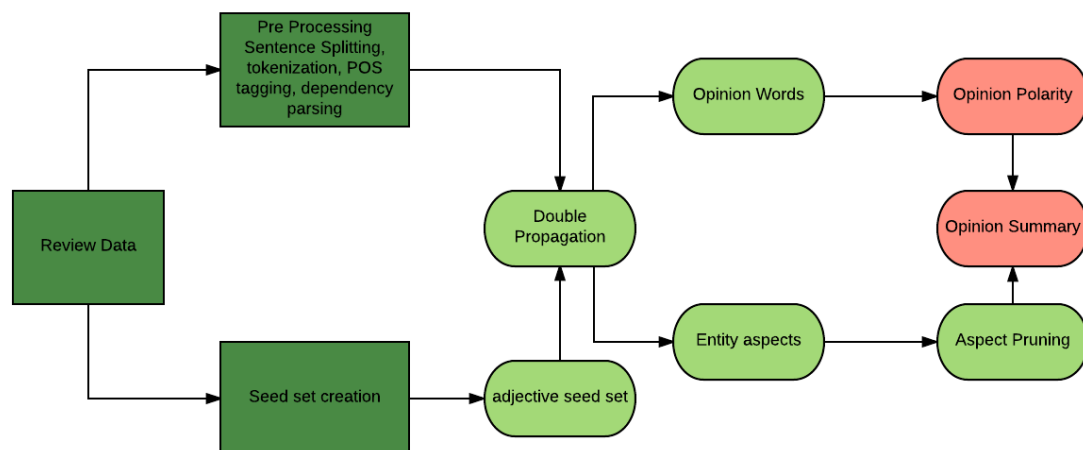
3.4 Data

The dataset used for implementation and testing is pre-annotated customer review dataset of 9 products used by Liu, Ding and Yu in 2008. (Liu, Ding, Yu,2008). For implementation and testing the annotation was completely removed, but this information was saved and used for analysis.

3.5 System Design

The proposed system contains 3 main elements: Entity aspect and sentiment word extraction which is done by double propagation, creation of seed set of adjectives, and aspect pruning. To do this initial steps of reading the data into the system, and pre-processing the data had to be taken. The system is described on the diagram below, which shows how the elements work together.

Figure 3.1 – System Diagram



To implement a full sentiment analysis software, two additional steps, opinion polarity analysis and opinion summary production (colored red in the diagram) would be added.

The system is also described on the use case diagram below.

Figure 3.2 – Use case diagram

SENTIMENT ANALYSIS - USE CASE DIAGRAM

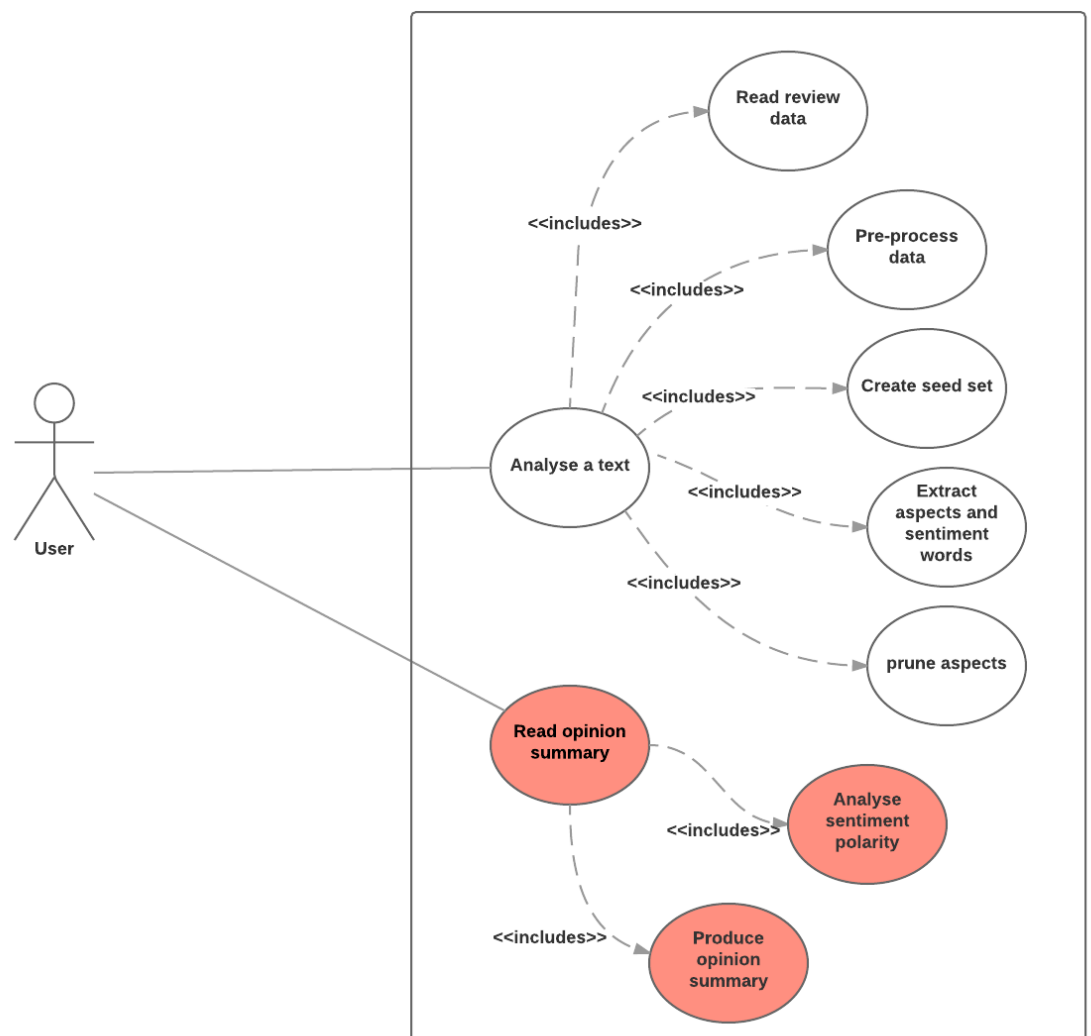


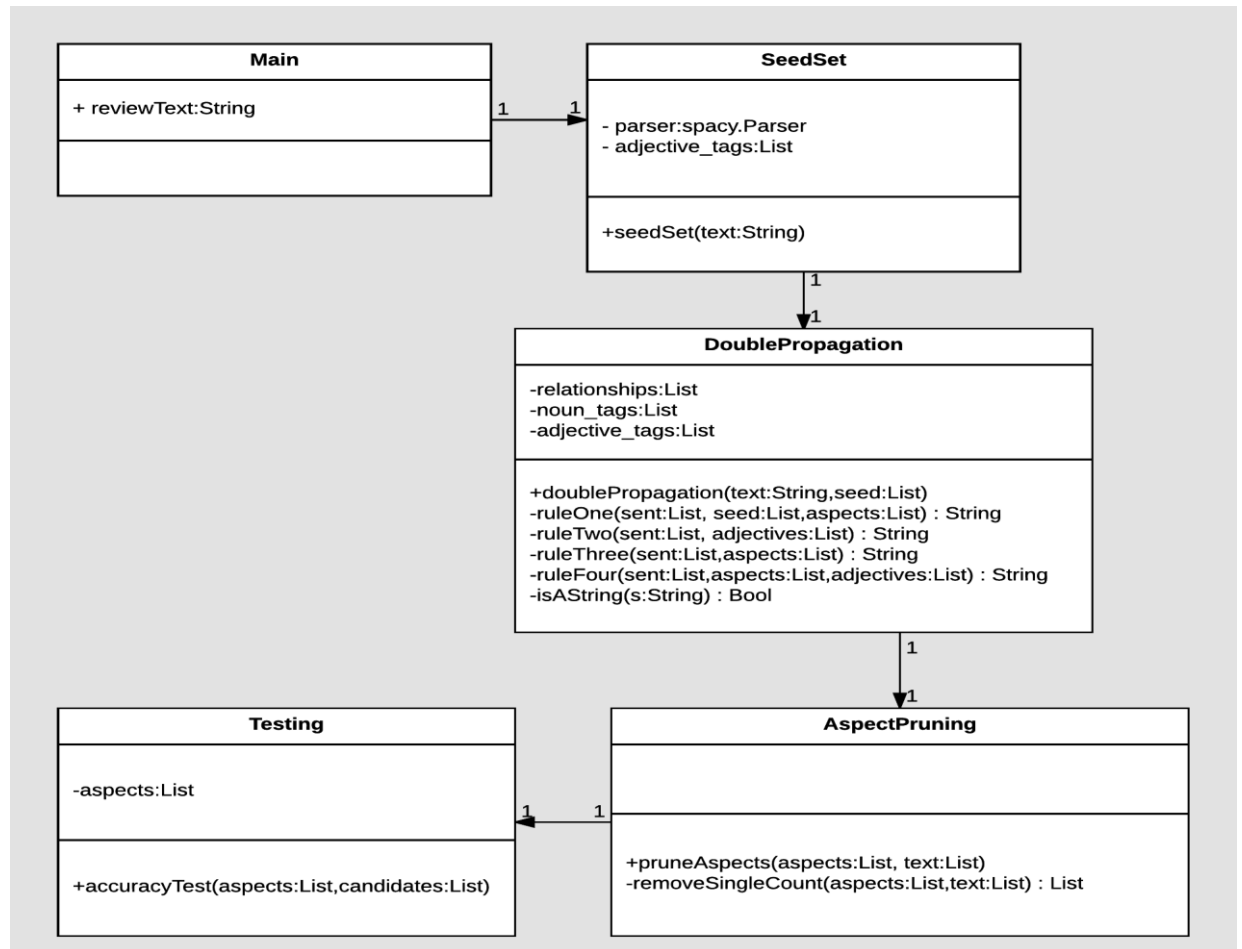
Table 3.1 Use Case Description

USE	DESCRIPTION
ANALYSE A TEXT	Actor/s: USER Pre-Condition: Data to analyse Main flow of events: <ol style="list-style-type: none">1. USER loads the review data into the program2. USER runs the main file of the program3. System pre-processes the data, creates the seed set, extracts opinion words and aspects and displays extracted aspects.4. USER reads the results from the screen
READ OPINION SUMMARY	Actor/s: User Pre-Condition: Text has been analysed Main flow of events: <ol style="list-style-type: none">1. System analyses the opinion polarity of extracted opinion words2. System Creates an opinion summary from analysed opinion words and aspects3. USER reads the summary from the screen

The use case Analyse text contains the steps to analyse the review text. It includes reading and pre-processing the data, creating the seed set, extracting aspects and sentiment words and pruning the aspects. The use case read opinion summary (colored red in the diagram) includes analysing the sentiment polarity of sentiment words and producing opinion summary. These steps are not included in this project.

Each of the system elements was implemented as a class, producing the following class diagram:

Figure 3.3– System Class Diagram

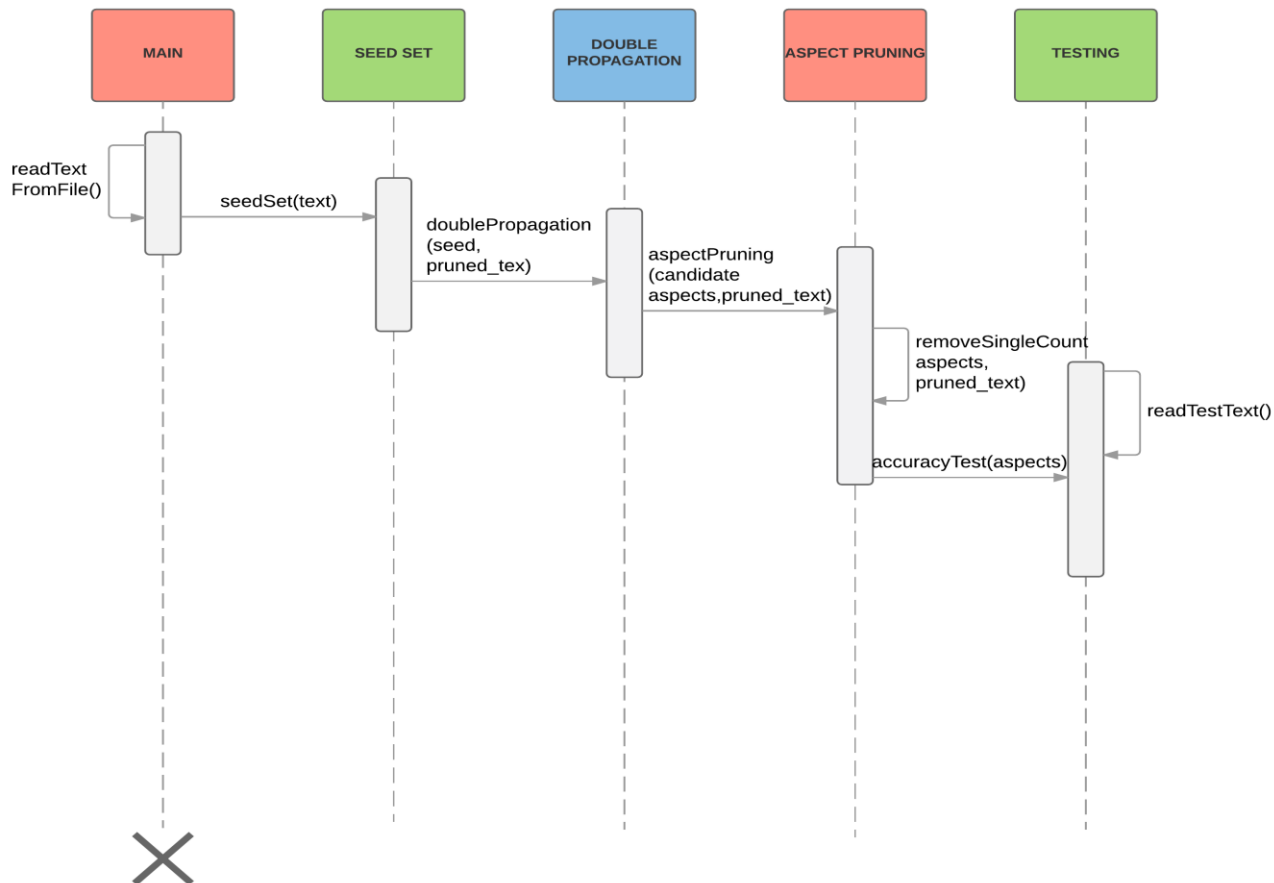


The main class, which reads from file and stores the original review text, calls an instance of the seedSet class with the review text. Seed set class contains methods to parse the text and create the seed set. DoublePropagation class is called by the seedSet class with the seed set and the parsed text. The class contains the double propagation algorithm method, and each of the extraction rules implemented as separate methods. AspectPruning class is then called by the DoublePropagation with the list of candidate aspects. This class contains methods to prune false positives from the aspect list. Testing class is then called by AspectPruning. Testing class contains the list of real aspects on the review text, and method to test how accurately the aspect set produced matches the real aspects list.

How these 5 classes work together can be seen from the sequence diagram below:

Figure 3.4 – Aspect Extraction Sequence Diagram

ASPECT AND SENTIMENT WORD EXTRACTION



First, the main reads the text file containing product reviews. The file is passed to seedSet, which parses the review text string into a list of tokens. From these tokens seed set is created using Frequency Distribution of adjectives. The seed set and the full list of tokens is passed to double propagation, which extracts a list of candidate aspects and list of sentiment words from the text. The list of candidate tokens together with the full token list are passed to aspect pruning, where false positives are removed from the aspect list. The testing class then reads the list of expected aspects from a text file, and compares the

aspect list produced to the expected list, printing out the results. The design and implementation of all these steps is discussed in detail below.

3.5 Summary

In this chapter the desing of the proposed system was detailed. The software used to implement the system, and the methodology used were both discussed. The design of each element of the sentiment analysis system was discussed and shown through UML diagrams.

4.Implementation

In this chapter the implementation of the system as designed in chapter 3 will be discussed. The Implementation of each part of the system, as shown in diagram 3.1, will be discussed in detail below.

4.1 Pre – Processing

The acquired dataset is pre- tagged to annotate entity aspects, opinion words, beginning of sentences and beginning of reviews. The information of these tags is used for testing, so all words tagged as aspects are saved for later use. From the rest of the data all the tags are manually removed, and the cleaned text is saved into a set of 4 .txt files, each containing the cleaned text of product reviews of one product.

4.2 Adjective Seed set Creation

It was hypothesised, that a set of the most used adjectives in the data would perform better as a seed set than a seed set acquired from outside sources. To acquire the adjectives, the frequency distribution of words POS tagged as adjectives was used.

The seed set creation works by the following pseudocode:

```
For all tokens in parsed text:  
  If token.POS in adjective_tags and token is not stopword, is not out of vocabulary  
  and is not punctuation:  
    Add token lemma to adjectives  
FrequencyDistribution(adjectives)  
From FreqDist(adjectives) take most common(5)  
For i in mostCommon:  
  i = double(string,dist)  
  seedSet add string
```

The method takes a string of text, parses it and produces a list of most frequent adjective lemmas in the text as well as a list of all tokens in the text. Both of which are passed on to double propagation.

4.2.1 Parsing

The whole text was translated into Unicode and parsed using spaCy parser.

The parser produces a list of tokens, each of which contain information on the POS tag, lemma, dependency and sentence boundary information of the token. One token is one word, whitespace, number or special character such as “.” Or “!” in the text.

The picture below shows the structure of spaCy token.

```
cdef struct TokenC:
    const LexemeC* lex
    uint64_t morph
    univ_pos_t pos
    bint spacy
    int tag
    int idx
    int lemma
    int sense
    int head
    int dep
    bint sent_start

    uint32_t l_kids
    uint32_t r_kids
    uint32_t l_edge
    uint32_t r_edge

    int ent_iob
    int ent_type # TODO: Is there a better way to do this? Multiple sources
    hash_t ent_id
```

(spaCy, n.d)³

³ spaCy, n.d. *Understanding spaCy's data model*. [Online] Available at: <https://spacy.io/docs/usage/data-model> [Accessed 22 April 2017]

The whole parsed document contains the vocabulary information of the document. For this project the spaCy English vocabulary was downloaded. The vocabulary information contains a stop word list, and a function to test whether a token text is out of vocabulary (is not a proper English word). To prevent false positives when creating the adjective seed list, stop words and out of vocabulary words were removed. Following the rules Qiu et al used in their implementation (Qiu et al, 2011), allowed POS tags in the adjective_tags were limited to JJ, JJS and JJR (adjective, comparative adjective and superlative adjective). For easier matching of words later, the lemmas of the adjectives were used. Lemmatization and lemmas as explained by Stanford NLP group means: “For grammatical reasons, documents are going to use different forms of a word, such as *organize*, *organizes*, and *organizing*. Additionally, there are families of derivationally related words with similar meanings, such as *democracy*, *democratic*, and *democratization*. In many situations, it seems as if it would be useful for a search for one of these words to return documents that contain another word in the set. The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance:

am, are, is \Rightarrow be

car, cars, car's, cars' \Rightarrow car

... *Lemmatization* usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma* . (Stanford, n.d). This ensures that words such as pretty and prettier are not extracted as separate words, but as one word.

4.2.2 Frequency Distribution

Frequency Distribution refers to the frequency of a word appearing in a text. For the seed set 5 adjective words with the highest frequency distribution were selected. Frequency Distribution was calculated by first finding the lemmas of all adjectives in the text, and then calculating the frequency of each lemma in the set of all tokens in the text.

The calculation produced a list of tuples (string, dist) from which the strings were extracted.

4.3 Double Propagation Algorithm

The seed set method passes the list of adjectives and a list of all tokens from the text to the double propagation method. As stated above, each token contains information of the sentence boundaries of the sentence that token belongs to. From this information, each sentence is produced, and all sentences in the text are iterated through token by token.

If a sentence contains a noun that is not in the aspect list or an adjective that is not in the adjective list, the sentence is passed on to the extraction methods.

This process is repeated until no more aspects or adjectives can be extracted. The extraction rules and the dependency tags used to extract words, as well as the design and functionality of the algorithm are discussed in detail below.

4.3.1 Algorithm overview

The design of the algorithm follows the original pseudocode by Qiu et al. (Qiu et al, 2011)

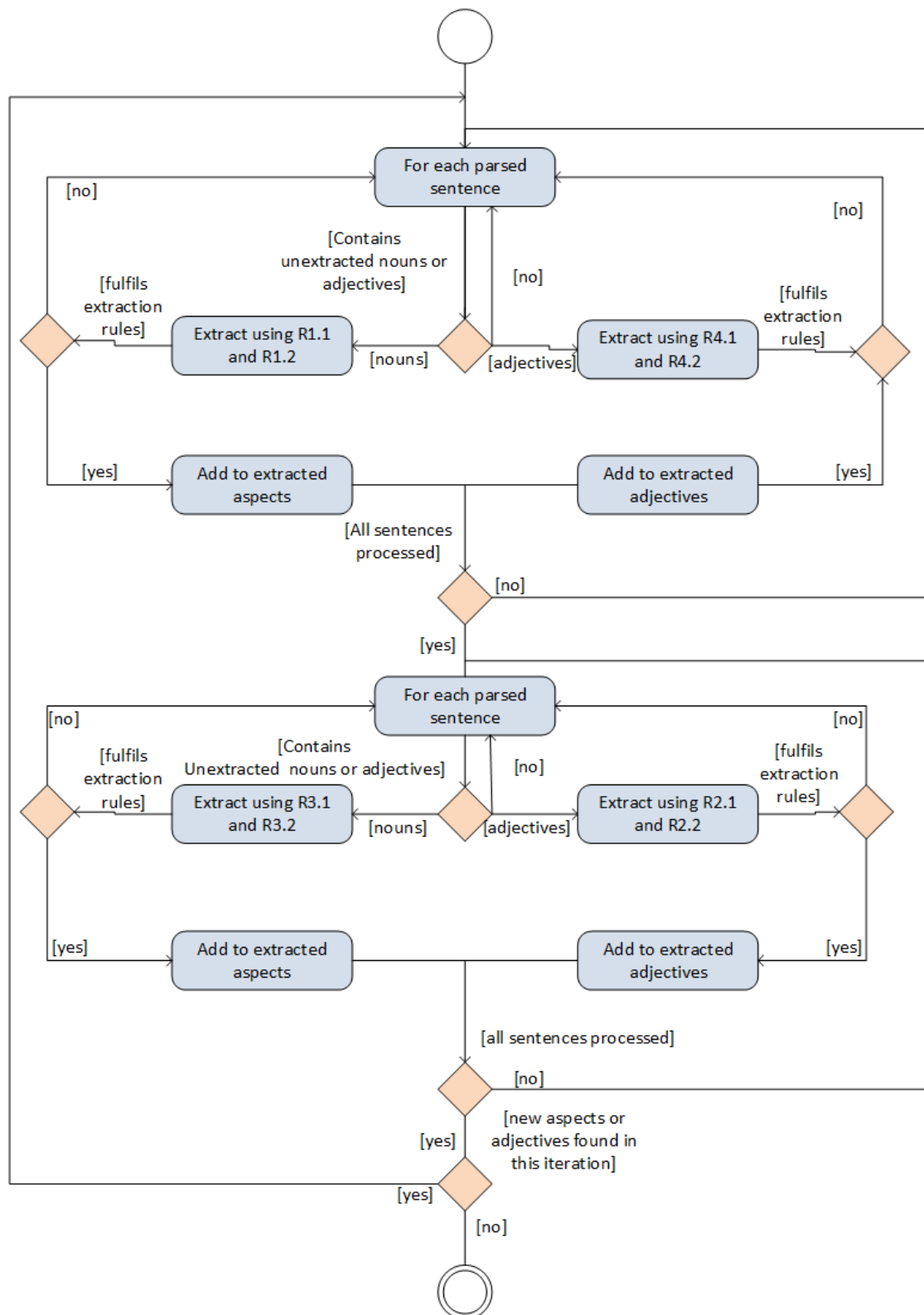
```
Input: Opinion Word Dictionary { O }, Review Data R
Output: All Possible Features { F }, The Expanded Opinion Lexicon { O-Expanded}
Function:
1. { O-Expanded } = { O }
2. { F } =  $\emptyset$ , { O } =  $\emptyset$ 
3. for each parsed sentence in R
4.   if(Extracted features not in { F })
5.     Extract features { F } using R11 and R12 based on opinion words in { O-Expanded}
6.   endif
7.   if(Extracted opinion words not in { O-Expanded})
8.     Extract new opinion words { O } using R41 and R42 based on opinion words in { O-Expanded}
9.   endif
10.  endfor
11. Set { F } = { F } + { F }, { O-Expanded } = { O-Expanded } + { O }
12. for each parsed sentence in R
13.   if(Extracted features not in { F })
14.     Extract features { F } using R31 and R32 based on features in { F }
15.   endif
16.   if(Extracted opinion words not in { O-Expanded})
17.     Extract opinion words { O } using R21 and R22 based on features in { F }
18.   endif
19. endfor
20. Set { F } = { F } + { F }, { O } = { O } + { O }
21. Set { F } = { F } + { F }, { O-Expanded } = { O-Expanded } + { O }
22. Repeat 2 till size({ F }) = 0, size({ O }) = 0
```

This pseudocode was followed throughout the method with small changes to amend it to the different parsing system.

Each extraction rule was implemented as a separate method, and as a unextracted nouns or adjectives are found, the sentence is passed to an extraction method, which returns the extracted aspect or an adjective, which are then added to `extracted_aspects` or `extracted_opinion` lists (O-Expanded and Fi in the original pseudocode).

The algorithm and the extraction methods work together as presented in the activity diagram on the next page.

Figure 4.1 – Double Propagation Activity Diagram



4.3.2 Dependency Tags

The original set of dependency tags used by Qiu et al had to be modified for this project. Due to differences in parsing technology (minipar vs. spaCy) and terminology used by the parsers, the set of relationships defined by Qiu did not match the available dependency relationship tags on spaCy parser. Qiu et al used the following relationships to extract Aspects using Opinion Words and Opinion words using Aspects. *mod*, *Pnmod*, *Subj*, *S*, *Obj*, *Obj2*, *Desc* And the following relationships to extract opinion words using opinion words or Aspects using other aspects: *conj* (Qiu et al, 2011)

Relationship tags *mod*, *pnmod*, *s*, *obj2* and *desc* do not exist in spaCy relationship tags, and direct translation between minipar dependency tags and spaCy(based on clearNLP) tags is not available. Using information provided by Qian et al on their paper *Improving Opinion Aspect Extraction Using Semantic Similarity and Aspect Associations*, where the team implemented Qiu's algorithm using Stanford parser, the relationship tag list was translated to the following: (*amod*, *prep*, *nsubj*, *csubj*, *xsubj*, *dobj*, *iobj* and *conj*). (Qian et al, 2016). SpaCy's dependency tags are still not always the same as the tags used by Stanford. The dependency parser of spaCy is based on clearNLP project, so using their documentation *Guidelines for the Clear Style - Constituent to Dependency Conversion* by Jinho Choi and Martha Palmer (Choi, Palmer, 2012) the Stanford tag list was compared to the clearNLP tag list. In this case, all the tags stayed the same, so the list of relationships used for this project is as follows:

Where in italics is the head of the relationship, and underlined is the child.

Table 4.1 – Dependency Tags

Tag	Explanation	Example sentence
Amod	Adjectival Modifier	"A <i>beautiful</i> <u>girl</u> "
prep	Prepositional Modifier	"Please <i>put</i> your coat <u>on</u> the table"
nsubj	Nominal Subject	" <u>She</u> and I <i>came</i> home together"
csubj	Clausal Subject	"Whether she <i>liked</i> me doesn't <u>matter</u> "
xsubj	Open Clausal Subject	"She seemed to like the hat"
dobj	Direct Object	"She <i>bought</i> me <u>these books</u> "
obj	Indirect Object	"She <i>bought</i> <u>me</u> these books"
conj	Conjunct	"John, Mary and <u>Sam</u> "

(Choi, Palmer, 2012).

Following the Qiu et al implementation, POS tags JJ, JJS and JJR are used to extract adjectives and NN,nd NNS are used to extract noun. (Qiu et al, 2011)

4.3.3 Extraction Rules

The dependency rule list determined above was used together with 8 extraction rules to extract opinion words and aspects. The extraction rules are used to extract aspects and opinion words four different ways: Extracting aspects using opinion words, extracting aspects using other aspects, extracting opinion words using aspects, and extracting opinion words using other opinion words. The rules are based on the Qiu algorithm, but due to the different nature of the parsing system used, some of the rules have been adapted. The original set of rules is as follows:

In the table: **{A}**,**{AS}** Stand for the set of extracted adjectives words or aspect **POS(A)**, **POS(AS)** – The point of speech tag of adjective or an aspect **(A)-Deb** , **(AS)-Deb** – Name of the dependency relationship **{JJ}** – Set of point of speech tags allowed for opinion words **{NN}** – set of point of speech tags allowed for Targets **{REL}** – set of allowed dependency tags for relationships

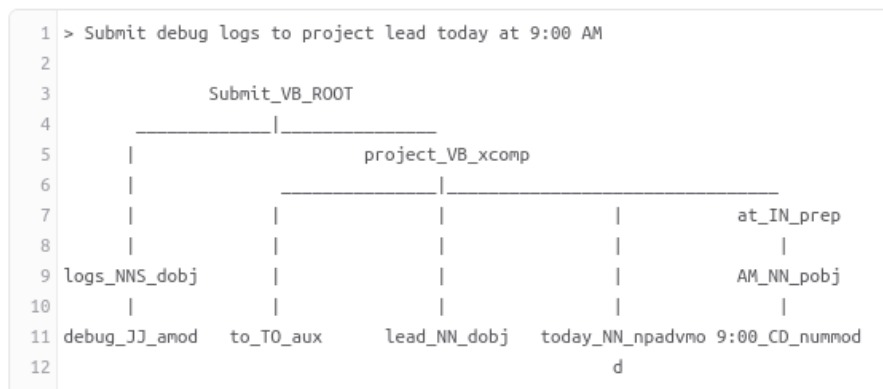
Table 4.2 – Extraction Rules

Rule	Observations	Extracts	Example
R1.1	$A \rightarrow A\text{-Deb} \rightarrow AS$ s.t A-Deb in {REL} and POS(AS) in {NN}	AS	“The phone has a <u>good screen</u> ” (good -> mod ->screen)
R1.2	$A \rightarrow (A)\text{-Deb} \rightarrow H \leftarrow AS\text{-Deb} \leftarrow AS$ s.t A in {A}, (A)-deb and (AS)-deb in {REL} POS(AS) in {NN}	AS	“iPod is <u>the best</u> mp3 player!” (best ->mod->player<-subj <- iPod)
R2.1	$A \rightarrow (A)\text{-Deb} \rightarrow AS$ s.t AS in {AS}, A-Deb in {REL}, POS(A) in {JJ}	A	As in R1.1 but good instead of screen extracted as opinion word
R2.2	$A \rightarrow (A)\text{-Deb} \rightarrow H \leftarrow (AS)\text{-Deb} \leftarrow AS$ s.t AS in {AS}, (A)-deb and (AS)-deb in {REL}, POS(A) in {JJ}	A	As in R1.2 but the best extracted as opinion word
R3.1	$AS1 \rightarrow (AS1)\text{-Deb} \rightarrow AS2$ s.t AS2 in {AS}, (AS1)-Deb == ‘conj’, POS(AS2) in {NN}	AS1	“The phone comes with a <u>screen protector</u> and <u>a case</u> ” (screen protector -> conj -> a case)
R3.2	$AS1 - (AS1)\text{-Deb} \rightarrow H \leftarrow (AS2)\text{-Deb} \leftarrow AS2$ s.t AS1 in {AS}, (AS1)-Deb == (AS2) –Deb, POS(AS2) in {NN}	AS2	“iPhone’s <u>camera</u> has a good <u>picture quality</u> ” (Picture quality -> obj -> has <- subj <-camera)
R4.1	$A1 \rightarrow (A1)\text{-Deb} \rightarrow A2$ s.t A2 in {A}, A2-Deb == ‘conj’, POS(A1) in {JJ}	A1	“The <u>camera</u> is great and <u>easy to use</u> ” (easy -> conj -> great)
R4.2	$A1 \rightarrow (A1)\text{-Deb} \rightarrow H \leftarrow (A2)\text{-Deb} \leftarrow A2$ s.t A1 in {A}, (A1)-Deb == (A2)-Deb, POS(A2) in {JJ}	A2	“If you are looking for <u>cheap</u> , <u>nice</u> smartphone, get Sony Xperia” (nice -> mod -> smartphone <- mod <- cheap)

(Qiu et al, 2011)

4.3.4 Dependency Parsing

Dependency parsing, which is the cornerstone of the algorithm, refers to analysing the syntactic structure of a sentence, and in the case of the spaCy parser used, building a parse tree, which stores the relationships between words in a sentence. The picture below shows the structure of a parse tree for the sentence “Submit debug logs to project lead today at 9:00 AM”



(Kadam, n.d)⁴

spaCy parses text so that each token has a head, and stores its dependency to its head. (In the example above the head of token log is Submit, and its dependency to its head is dobj) each token also can have one or more children. (In the example above to, lead, today and at are all children of project). Following these dependency archs from token to its head and its children is how the algorithm finds two words that fulfil the extraction rules discussed above. For example, the review sentence “This is great camera” parses like this:

⁴ Shirishkadam, n.d. *Dependency Parsin NLP*. [Online] Available at: <https://shirishkadam.com/2016/12/23/dependency-parsing-in-nlp/> [Accessed 22 April 2017]

Table 4.3 – Dependency Parsing Example

Word	Dependency	Head
This	nsubj	is
is	ROOT	
great	amod	camera
camera	attr	is

Where word “this” is a child of word “is”, and the arch between them is called nsubj, similiarly “great” is a child of “camera” and the arch between them is called amod.

If adjective “great” was extracted to the seed set, using rule R1.1 the aspect Camera could be extracted from the sentence following that great is an adjective, in the seed set, and it’s dependency is in the relationships list, and the POS tag of camera is NN. The details of design and implementation of the rules is discussed below.

4.3.5 Extraction Rule Methods – Design and Implementation

Each rule pair (1.1 and 1.2, 2.1 and 2.2 and so on) is implemented as a method. The methods can be found in the `DoublePropagation.py` file of the delivered program.

Rule 1.1

$$A \rightarrow A-Deb \rightarrow AS$$
$$s.t \text{ A-Deb in } \{REL\} \text{ and } POS(AS) \text{ in } \{NN\}$$

Which written out means that if an adjective, which is in the seed list, is in a relationship with a noun and the relationship is on the list of allowed relationships, the noun is extracted.

Which became:

```
#extraction rule 1
def ruleOne(sent, seed, aspects):
    #iterate through tokens in sent
    for token in sent:
        #if token is an adjective, in seed set and it's relationship is in relationships
        if token.tag_ in adjective_tags and token.lemma_ in seed and token.dep_ in relat
            #further if token's head is a noun and token's head not in aspects
            if token.head.tag_ in noun_tags and token.head.lemma_ not in aspects:
                #if token head is a proper word
                if token.head.is_oov == False and token.head.is_stop == False:
                    #return token's head
                    return token.head.lemma_ |
```

Rule 1.2

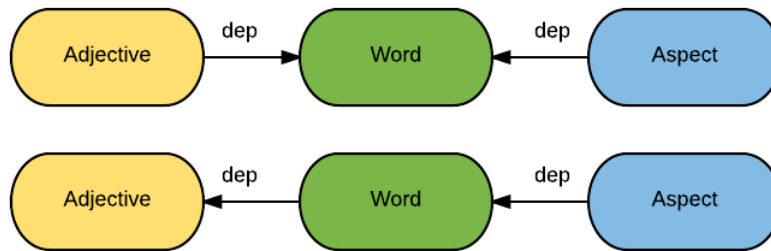
$$A \rightarrow (A)\text{-Deb} \rightarrow H \leftarrow AS\text{-Deb} \leftarrow AS$$
$$s.t\ A\ in\ \{A\},\ (A)\text{-}deb/\ (AS)\text{-}deb\ in\ \{REL\}\ POS(AS)\ in\ \{NN\}$$

Which translates to: If an adjective, which is already extracted and noun are both dependent on the same word, and they depend on it with a relationship in the relationships

list, extract the noun.

Due to the directionality of the dependency tags, this became two statements: If adjective's head's child is a noun. If a words head is adjective, and word's child is a noun.

Diagram 4.2 – Extraction rule 1.2



In the diagram arrows represent dependencies between words, and arrow points from the child to the head.

As code this looks like:

```
if token.tag_ in adjective_tags and token.lemma_ in seed and token.dep_ in relationships:
    for child in token.head.children:
        if child.tag_ in noun_tags and child.lemma_ not in aspects and child.dep_ in relationships:
            if child.is_oov == False and child.is_stop == False:
                return child.lemma_

if token.head.tag_ in adjective_tags and token.head.lemma_ in seed and token.dep_ in relationships:
    for child in token.head.children:
        if child.tag_ in noun_tags and child.lemma_ not in aspects and child.dep_ in relationships:
            if child.is_oov == False and child.is_stop == False:
                return child.lemma_
```

Rules 2.1 and 2.2

Rules 2.1 and 2.2 are used to extract new adjectives using already extracted aspects. The rules are the same as rules 1.1 and 1.2, but the known word being the noun and extracted word being the adjective.

For example, when rule 1.1 code is:

if token.tag_ in adjective_tags and token.lemma_ in seed and token.dep_ in relationships: if token.head.tag_ in noun_tags and token.head.lemma_ not in aspects:

Code for rule 2.1 becomes:

if token.tag_ in adjective_tags and token.lemma_ not in adjectives and token.dep_ in relationships: if token.head.tag_ in noun_tags and token.head.lemma_ in aspects:

Rule 2.1 code is similiarly derived from rule 1.2

Rules 3.1 and 3.2

Rules 3.1 and 3.2 are used to extract aspects using already extracted aspects.

3.1

AS1 -> (AS1)-Deb -> AS2 s.t AS2 in {AS}, (AS1)-Deb == 'conj', POS(AS2) in {NN}

Means finding a noun which has not been extracted, and if it's dependency tag is 'conj' and it's head is another noun which has been extracted, extract the noun.

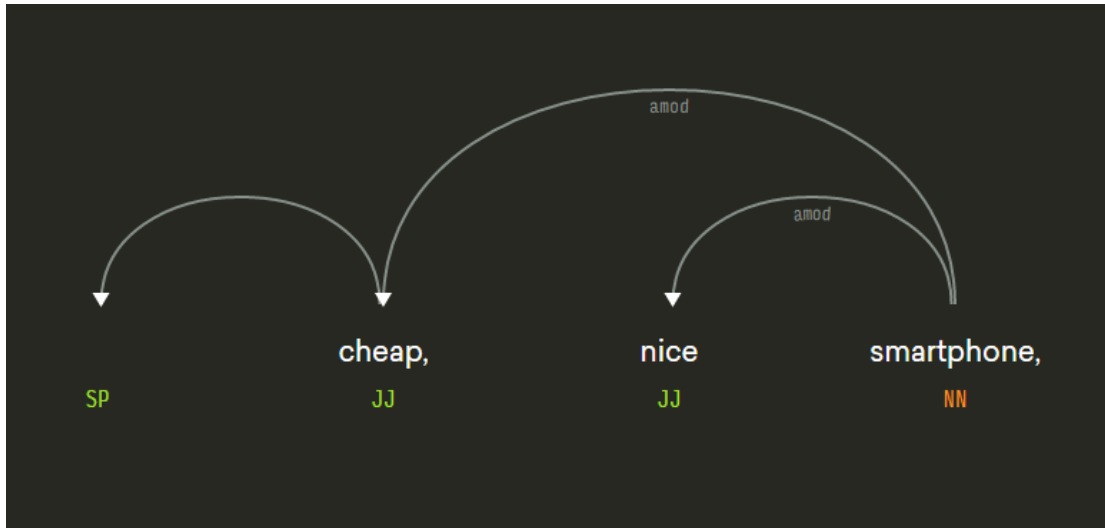
As a code, it functions similiarly to rules 1.1 and 2.1, with additional if token.dep_ == 'conj' instead of if

Token.dep_ in relationships.

3.2

AS1 - (AS1)-Deb -> H <- (AS2)-Deb <- AS2 s.t AS1 in {AS}, (AS1)-Deb == (AS2) - Deb, POS(AS2) in {NN}

With this rule and rule 4.2 the concept of equivalent dependencies is used. In the original article Qiu states that: ““=” represents the same or equivalent (Here equivalent specifically means mod is the same as pnm, and s or subj is the same as obj).”(Qiu et al, 2011). The spaCy taglist, as stated above, does not contain tags mod, pnm, s, subj or obj, so the rule had to be modified. Through parsing example sentences using displacy, spaCy parsing visualizer[3] it was determined that “pnm is the same as mod” becomes amod == amod.



(Displacy, n.d)⁵

S, SUBJ and OBJ are with spaCy replaced by NSUBJ, CSUBJ, DOBJ and OBJ.

With these it was determined that the equivalencies are: NSUBJ & CSUBJ == DOBJ & OBJ. So rule 3.2 becomes: If word is a noun, it's dependent on a word H with a same or equivalent relationship as another noun, and the other noun is not yet extracted, extract the other noun. As a code this looks like:

```
if token.tag_ in noun_tags and token.lemma_ in aspects and token.dep_ in equivalent:
    for child in token.head.children:
        if child.tag_ in noun_tags and child.dep_ in equivalent and child.lemma_ not in aspects:
            if child.is_stop == False and child.is_oov == False:
                return child.lemma_
```

⁵ Displacy, n.d. *displaCy* [Online] Available at: <https://demos.explosion.ai/displacy/> [Accessed 22 April 2017]

Rules 4.1 and 4.2

Rules 4.1 and 4.2 are used to extract adjectives using other adjectives, and they work similarly to rules 3.1 and 3.2

Rule 4.1

$A1 \rightarrow (A1)\text{-}Deb \rightarrow A2$

s.t $A2 \in \{A\}$, $A1\text{-}Deb == conj$, $POS(A1) \in \{JJ\}$

This is the same as rule 3.1, but the word and the head are adjectives.

Rule 4.2

$A1 \rightarrow (A1)\text{-}Deb \rightarrow H \leftarrow (A2)\text{-}Deb \leftarrow A2$

s.t $A1 \in \{A\}$, $(A1)\text{-}Deb == (A2)\text{-}Deb$, $POS(A2) \in \{JJ\}$

This is the same as rule 3.2, but the word and the child are adjectives. Here again we have the concept of equivalent dependencies, and as stated above, for adjectives it was determined that equivalencies mean $amod == amod$.

Each rule couple (1.1 and 1.2, 2.1 and 2.2 and so on) was implemented as a separate method. The algorithm itself takes as an input the list of all tokens in the text, and iterates over it sentence by sentence. During the first loop, the sentences are iterated over token by token, until a token which POS is noun or an adjective, and it is not yet in the aspect list or the adjective list, is encountered. The sentence is then sent to the extraction methods 1 and 4, which implement rules 1.1, 1.2, 4.1 and 4.2. Each extraction method iterates over the tokens of the sentence, and returns a word if one is extracted. Details of each rule are provided above. Returned tokens are saved into `extracted_adjectives` and `extracted_aspects` lists. When each sentence has been processed, the `extracted_adjectives` are added to `adjectives` and `extracted_aspects` are added to `aspects`. In the second loop, each sentence is processed again, and rules 2.1, 2.2, 3.1 and 3.2 are used to extract more adjectives and aspects using the adjectives and aspects extracted in loop 1. At the end of

loop one it is checked if `extracted_aspects` and `extracted_adjectives` are empty, which would mean no more aspects or adjectives have been found. Both loops are repeated until no more words can be extracted. When no more words can be found, the extracted aspects are passed onto pruning and testing.

4.4 Aspect Pruning

Some aspect pruning elements were already implemented throughout the extraction rules. It was noticed, that stopwords are rarely important aspects and including them to aspect list produces a lot of false positives, so all stopwords are removed before aspects are added to aspect list. This is done by utilizing the vocab data of the text. The `is_stop` method also used when implementing the seed set method was used.

To remove misspelled or other wrong words, the `is_oov` (is out of vocabulary) method of vocab was used. It was also noticed, that sometimes, writing mistakes such as ‘ii’ or ‘t’ are passed on as nouns, so the program removes all nouns less than 2 characters long. It is also recognized that no real aspect noun in English is shorter than 3 characters long, so no important aspects could be missed this way.

It was further noticed, that a clear majority of the important aspects on the text are talked about various times and throughout different reviews. The aspect candidates which are only mentioned once in the whole dataset are often writing mistakes or non-aspects such as “photobugs” or “camena”. So, it was decided to remove all aspect candidates which are mentioned only once in the whole dataset.

This was achieved by finding the frequency distribution of the candidate aspects in the original text, and removing the ones which appeared only once.

4.5 Summary

In this chapter the implementation of the system has been discussed. Parts 4.1 and 4.2 showed the pre-processing in detail and discussed how the seed set creation method was implemented. Details of dependency parsing were also discussed. 4.3 showed the details of the double propagation algorithm. The extraction rules, dependency parsing and the dependency tags used were discussed in detail. The implementation of each rule was also shown in part 4.3.5. The implementation of aspect pruning methods were discussed in part 4.4

5. Testing

This section contains the testing results of the project. Two kinds of tests were performed. First, Functionality testing was performed to ensure that the extraction rules perform as required. For this example, sentences known to contain aspects or adjectives were passed to the algorithm. The results of this can be found from table 5.1. The algorithm extracted the right words 100% of the time, ensuring that all the rules are working as required.

For final testing, the data described in section 3.1.3 was used. 4 datasets each containing reviews of one product were used. Each dataset contained 100 sentences, which tags were removed from as described in section 3.2. From each original, tagged dataset the nouns tagged as aspects were saved as a comparison list with certain restrictions: As this system only considers explicit aspects, only aspects mentioned in the text were considered. Only aspect words which are nouns were considered, as the system only extracts noun aspects.

To test whether the added seed set creation method affects the performance of the algorithm, each dataset was also tested without the seed set creation method. For these tests a seed set of most common adjectives in the English language derived from Oxford Dictionary⁶ was used. The results these tests are in section 5.2 tables. 5.2 – 5.5.

Section 5.3 contains summarisation of the testing results.

5.1 Functionality Testing

⁶ Oxford living dictionaries, n.d. *What can the Oxford English Corpus tell us about English Language?* [Online] Available at: <https://en.oxforddictionaries.com/explore/what-can-corpus-tell-us-about-language> [Accessed 22 April 2017]

The table below shows the results of functionality tests performed. First column shows the sentence used, and following columns show the word(s) to be extracted from the sentence, rule the test is testing and the word(s) actually extracted.

Table -5.1 Functionality results testing of the double propagation algorithm.

Sentence	To be extracted	Rule	Extracted
“The phone has a good screen”	Screen (aspect) great (adjective)	1.1	Screen great
“The phone is the weirdest smartphone”	smartphone	1.2	smartphone
“It has good keypad. Does the phone come with with keypad and microphone?”	First round – keypad 2 nd round - microphone	1.1 -> 3.1	Keypad microphone
“The phone has a nice battery.”	nice	2.1	nice
“The phone is good and easy to use.”	easy	4.1	easy
“It is good, sexy phone”	sexy	4.2	sexy
“The phone is the weirdest smartphone”	weird	2.2	weird

5.2 Accuracy Testing

Each testing table shows the results of accuracy tests done on one dataset. Datasets were described above. First column shows the aspects present in the set, the following columns showing which ones of them were extracted with and without the seed set creation method, and the false positives extracted.

Table 5.2 – Dataset 1 Apple iPod

Correc aspects	Extracted With seed set	Incorrect aspects extracted	Extracted without seed set	Incorrect aspects extracted
feature	x	device		thing
interface		buy		problem
game	x	thing	x	
Sound				
Battery				
Photo	x			
quality	x			
design	x			
player	x			
iPod	x		x	
10	8	3	2	2

Table 5.3 – Dataset 2 Canon100

Aspects in Set	Correct aspects with seed set	False Positives with seed set	Correct aspects without seed set	False positives without seed set
size	x	Thing		Thing
picture	x	End	x	Problem
camera	x	Card	x	Card
quality	x	Choice		choice
battery	x		x	
zoom	x			
flash	x		x	
autofocus				
screen				
viewfinder				
mode	x		x	
image	x			
12	9	4	5	4

Table 5.4 – Dataset 3 Apex DVD player

Aspects	Extracted with seed set	False Positives with seed set	Extracted Without seed set	False Positives Without seed set
dvd		Thing		Week
player	x	Review	X	Run
button	x	Problem		Review
layout	x	movie	X	Problem
feature	x			Month
rewind				Place
video	x		X	Electronics
Format			X	day
price	x			
audio	x		X	
control	x			
quality	x		x	
value	x		X	
picture	x		X	
zoom				
signal	x			
Display	x			
18	13	4	8	8

Table 5.5 Dataset 4 Nokia mobile phone

Aspects in the data set	Aspects extracted with the seed set	False positives with the seed set	Aspects extracted without the seed set	False positives without the seed set
Phone	x	Life	X	Life
Signal	x	pim	X	Strength
Reception	x	Combination	X	Combination
Feature	x	price	X	One
Radio	x	one	X	Year
Headset	x		X	Care
Memory				panasonic
Csr	x		X	
Plan			X	
Menu	x		X	
Nokia	x		X	
Battery				
Size	x			
Game	x		X	
Sound	x			
Quality				
Earpiece				
Speakerphone				
Screen				
Button				
service	x		X	
Volume	x		X	
22	15	5	13	7

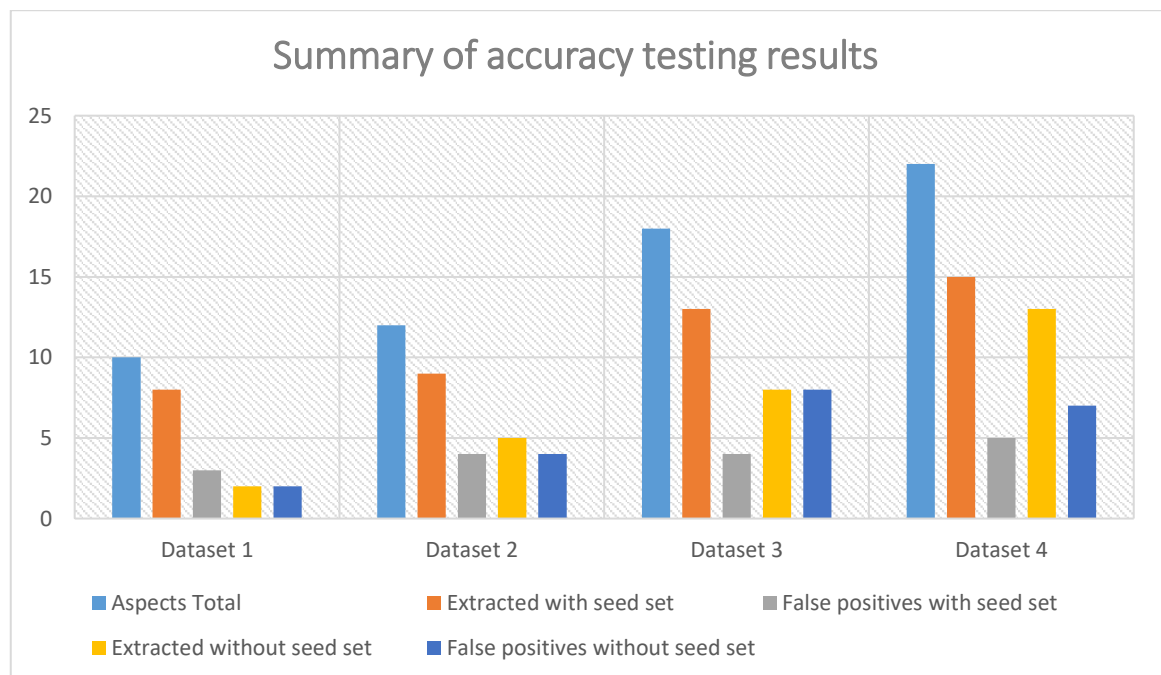
5.4 Summary

From the functionality testing results on table 5.1 we can see, that each extraction rule works as predicted extracting the right word 100% of the time.

The accuracy testing results, as shown in tables 5.2 to 5.5, show that every time more correct aspects were extracted when the seed set method was used than when a seed set acquired elsewhere was used. The algorithm, when using the seed set method, also extracted less false positives every time.

These results are summarised in chart 5.1 below, which shows the amount of real and false aspects extracted with and without the seed creation method in every dataset.

Figure 5.1 – Summary of accuracy testing results.



The testing results will be discussed in detail in chapter 6 below.

6. Discussion

In this chapter the testing results from accuracy tests done in chapter 5 will be discussed in detail. The accuracy and precision calculations derived from the testing will be discussed, and the effect of the seed set creation method on the performance of the algorithm will be detailed.

General restrictions of the algorithm and ideas on how the algorithm could be made more functional in more types of text will also be discussed below.

6.1 Accuracy and precision of the algorithm

This section contains calculations of accuracy and precision of each dataset (apple iPod, Canon 100, Apex DVD player and Nokia mobile phone) with and without the seed set (figure 6.1), and calculation of median accuracy and precision of the algorithm with and without the seed set (figure 6.2).

Precision was calculated using the formula: $P = TP/TP+FP$ where TP is the number of true positive aspects extracted, and FP is the number of false positives.

Accuracy was calculated using the formula: $A = (TP + TN) / (TP + TN + FP + FN)$

Where TP is the number of true positives, TN is the number of true negatives, which is derived from the number of different lemmas in the text, FP is the number of false positives and FN is the number of false negatives.

Figure 6.1 – Accuracy and Precision by Dataset

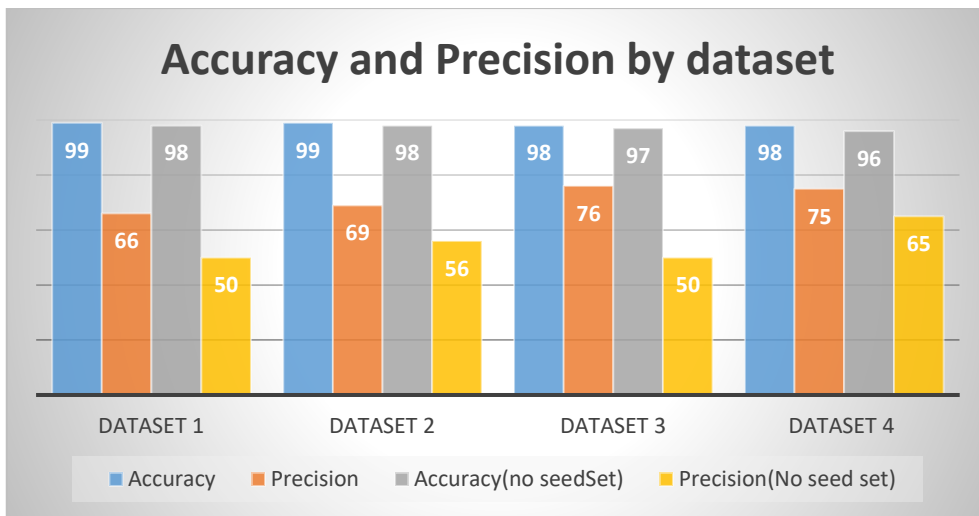
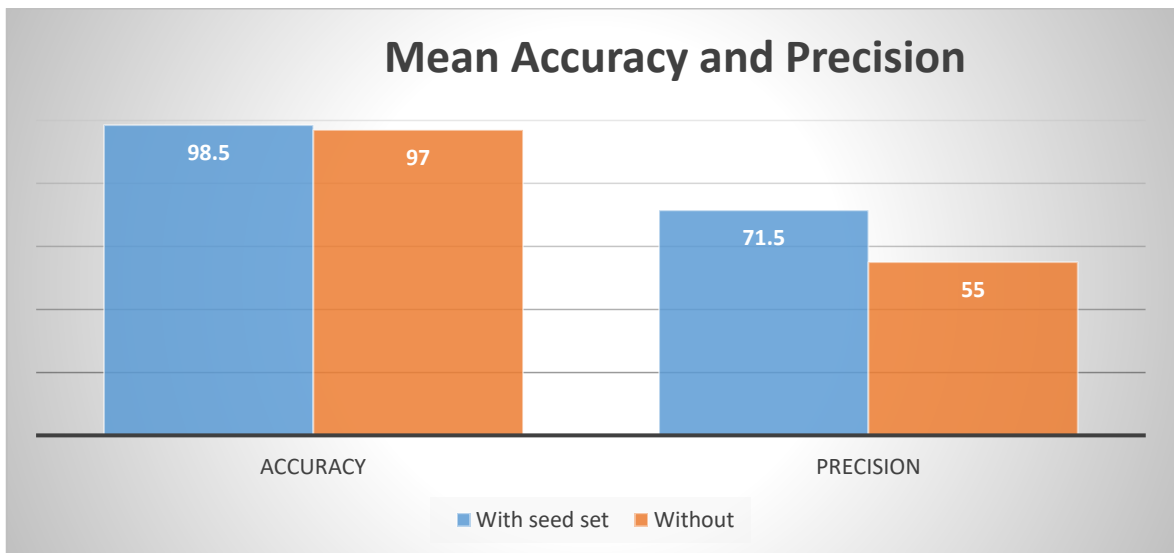


Figure 6.2 – Mean Accuracy and Precision



As can be seen from the figures above, the accuracy of the algorithm is great overall varying between 99 and 96%. The algorithm without the seed set has median accuracy of 97%, which is the same or even 1% better than the worse performance of the algorithm with the seed set creation. This shows that the seed set creation method does not affect the accuracy of the algorithm because with or without the seed set method the algorithm can accurately classify most words in the text as non-aspects.

However, the seed set creation method has a big effect on the precision of the algorithm. The mean precision of the algorithm with the seed set creation method is 71.5 % and without it the precision drops to 55%, showing that the new method raises the precision of the algorithm 16.5% in average.

As can be seen from the testing results, the algorithm performs well, and can extract correct aspect words from review text with high accuracy. It can also be seen, that the added seed set method functions well and produces significantly better precision when compared to a seed set acquired elsewhere.

6.2 Double propagation and different text domains

The extraction rules as proposed by Qiu et al perform well, especially if the language used is close to proper English grammar.

However, often, especially when data is acquired from the internet, reviews are written using relaxed language. It is assumed that these extraction rules would not be accurate with all texts, such as relaxed social media texts consisting emoticons and hashtags which also imply opinions and emotions. New rules could be added and rules could be revised to accommodate more kinds types of writing.

It is also true, that aspects or and opinions do not only appear as adjectives and nouns. Sometimes aspects can be verbs (“works well, performs and so on) or other words, and opinions can appear as adverbs or verbs (love it, hate it and so on). More research onto the computational linguistics of opinion data is needed to build a complete list of rules to extract these kinds of aspects and opinions.

Non-aspects can also appear in the text in the context of the rules, and be extracted as false positives. These can be mostly pruned out using methods discussed above. Other methods, such as rules to find the references to products of another company or

comparisons to other products can be implemented to prune out more false positives. Due to the limited scope of this project, these are left to future improvements.

6.3 Algorithm Replication Issues

Comparing the results achieved with this implementation of the algorithm to other implementations can be difficult. As Marrese-Taylor and Matsuo found out in their article “*Replication issues on syntax-based aspect extraction for opinion mining*” (Marrese-Taylor, Matsuo) for which they implemented Qiu’s algorithm, the journal articles describing these algorithms do not necessarily contain all the necessary information for replicating the algorithm exactly. This issue arises when the term “equivalent” dependencies had to be deducted, as described in part 3.6.5. Also, due to advances in software development the tools used by previous implementations might be outdated, such as minipar used by Qiu, and the parameters used with old technology might not completely match the parameters used by new software, such as the case of dependency tags being different in spaCy than in minipar.

6.4 Summary

The algorithm performs well, being able to extract most important aspects from each review texts. The addition of the seed set method also raises the precision of the algorithm significantly. It is assumed it would perform as well when similar texts would be analysed. However, more research into computational linguistics and the vocabulary and grammar used in different text domains might be needed to compile a full list of extraction rules for all different domains.

7. Conclusion

As discussed in section 6, the testing results from the algorithm were good. The addition of seed set creation method to the algorithm both functions well and raises the precision of the algorithm significantly. It does not however affect the accuracy of the algorithm, as the changes were found to be less than 1% between the algorithms.

It does diminish the need to acquire a seed set of adjectives from elsewhere, making the initial setup of the algorithm easier, and fulfilling the aim to make the semi-supervised algorithm fully unsupervised.

However, there are limitations on what kind of texts the algorithm can analyse. Aspects and opinion words not appearing as nouns or adjectives are missed by the algorithm. Also, text consisting more grammatically relaxed language and special characters such as emoticons might be harder to analyse with this algorithm.

The aims of the project, as set in section 1.1 were fully met.

Aim 1, to research existing entity aspect extraction methods, was met in the literature review on chapter 2. In chapter 2 unsupervised and supervised algorithms and their results were also compared, meeting aim 2 (Compare unsupervised and supervised aspect extraction methods and the results achieved with them).

Aim 3, to test the implemented algorithm with and without the seed set creation method and analyse the findings to determine whether it affects the performance of the algorithm, was met. The implementation of the algorithm can be found from chapters 3 and 4, and the testing done with and without the seed set method can be found from chapter 5.

The system could be implemented into a full sentiment analysis tool if sentiment polarity analysis of the extracted adjectives and an opinion summary production methods were added. These were left for future improvements.

References

- Bird, S., Klein, E. and Loper, E. (2009). *Natural language processing with Python*. 1st ed. Beijing: O'Reilly.
- Chen, C., Bu, J., Liu, B. and Qiu, G. (2009). Opinion Word Expansion and Target Extraction through Double Propagation. *Proc. 21st Int. Joint Conf. Artif. Intell.*, pp.1199–1204.
- Chua, T., Wang, M., Zha, Z. and Yu, J. (2011). Aspect Ranking: Identifying Important Product Aspects from Online Customer Reviews. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, [online] pp.1496-1505. Available at: <http://www.aclweb.org/anthology/P11-1150> [Accessed 5 Mar. 2017].
- Eisenstein, J., Ji, Y. and Bhatia, P. (2015). Better document-level sentiment analysis from RST discourse parsing. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. [online] Available at: <https://arxiv.org/pdf/1509.01599.pdf> [Accessed 4 Mar. 2017].
- Feldman, R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM*, [online] 56(4), pp.82-89. Available at: <http://cacm.acm.org/magazines/2013/4/162501-techniques-and-applications-for-sentiment-analysis/fulltext> [Accessed 4 Mar. 2017].
- Frasincar, F. and Schouten, K. (2015). Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*, [online] 28(3), pp.813 - 830. Available at: <http://ieeexplore.ieee.org/document/7286808/?reload=true> [Accessed 4 Mar. 2017].
- Gurevych, I. and Jakob, N. (2010). Extracting opinion targets in a single-and cross-domain setting with conditional random fields. *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-2010)*, [online] pp.1035-1045. Available at: <http://www.aclweb.org/anthology/D10-1101> [Accessed 5 Mar. 2017].

- He, Y. and Lin, C. (2009). Joint sentiment/topic model for sentiment analysis. *CIKM '09 Proceedings of the 18th ACM conference on Information and knowledge management*, [online] pp.375-384. Available at: <http://dl.acm.org/citation.cfm?id=1646003> [Accessed 4 Mar. 2017].
- Ho, H. and Jin, W. (2009). A novel lexicalized HMM-based learning framework for web opinion mining. *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. [online] Available at: <http://people.cs.pitt.edu/~huynv/research/aspect-sentiment/A%20novel%20lexicalized%20HMM-based%20learning%20framework%20for%20web%20opinion%20mining.pdf> [Accessed 5 Mar. 2017].
- Hoffmann, P., Wiebe, J. and Wilson, T. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. *HLT '05 Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, [online] pp.347-354. Available at: <http://dl.acm.org/citation.cfm?id=1220619> [Accessed 4 Mar. 2017].
- Jin, C., Ng, H., Felker, M., Chang, E., Bierhoff, K. and Scaffidi, C. (2007). Red Opal: Product-feature scoring from reviews. *Proc. 8th ACM Conference on Electronic Commerce.*, [online] pp.182 - 191. Available at: <https://pdfs.semanticscholar.org/d22b/1bcd1c525c9130ffa412ca9dd71ebbc2a2b1.pdf> [Accessed 4 Mar. 2017].
- Liu, B. (2012). *Sentiment analysis and opinion mining*. San Rafael, CA: Morgan and Claypool Publishers.
- Liu, B. and Hu, M. (2004). Mining and summarizing customer reviews. *KDD '04 Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, [online] pp.168-177. Available at: <http://dl.acm.org/citation.cfm?id=1014073> [Accessed 4 Mar. 2017].

- Liu, B., Labs, H., Lim, S. and Zhang, L. (2010). Extracting and ranking product features in opinion documents. *Proc. 23rd Int.Conf. Comput. Linguistics*, [online] pp.1462-1470. Available at: <http://www.aclweb.org/anthology/C10-2167> [Accessed 5 Mar. 2017].
- Marrese-Taylor, E. and Matsuo, Y. (2017). Replication issues in syntax-based aspect extraction for opinion mining. [online] Available at: <https://arxiv.org/pdf/1701.01565.pdf> [Accessed 16 Apr. 2017].
- McDonald, R. and Täckström, O. (2011). *Semi-supervised latent variable models for sentence-level sentiment analysis*. [online] Available at: <http://dl.acm.org/citation.cfm?id=2002848> [Accessed 4 Mar. 2017].
- P, R., R, S. and C, T. (2016). A comprehensive survey on aspect based sentiment analysis. *International Journal of Advanced Research in Computer Science and Software Engineering*, [online] 6(4). Available at: https://www.ijarcse.com/docs/papers/Volume_6/4_April2016/V6I4-0192.pdf [Accessed 4 Mar. 2017].
- Qian, L., Liu, B., Zhang, Y., Kim, D. and Gao, Z. (2016). Improving Opinion Aspect Extraction Using Semantic Similarity and Aspect Associations. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*.
- Reynar, J., Reis, G., Neylon, T., McDonald, R., Hannan, K. and Blair-Goldensohn, S. (2008). Building a Sentiment Summarizer for local service reviews. *Proc. Workshop NLP Inf.Explosion*.
- Stanford, n.d. Stanford. *Stemming and Lemmatization*. [Online] Available at: <https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html> [Accessed 26 April 2017]

Appendix

Please refer to the attached disk for an electronic copy of this report along with all the supplementary data used to create it:

- Electronic copies of the interim report, Poster presentation and Final Project
- Copy of the Poster originally produced in march can be seen below in appendix 1
- Interim Report; The initial outline of the project submitted at the beginning of the year can be seen below in appendix 2.
- Source code

The source code for the algorithm and the accompanying methods can be found from the included disk. The source code can be run from any commant prompt if Python 2 or 3 and spaCy have been downloaded.

The source code contains 5 python files: Main.py, seedSet.py, DoublePropagation.py, Parsing.py and Testing.py


The program can by run by running Main.py. The program will produce the results of one dataset.

- Datasets and Testing aspect lists.

Dataset used to test the program can be found on files iTest.txt, Canon100.txt, Apex.txt and Nokia.txt

Lists of aspects derived from the datasets used for testing can be found on files iTestTest.txt, CanonTest.txt, ApexTest.txt and NokiaTest.txt


1.0 Project Poster



Anglia Ruskin University

Entity Aspect Extraction using Double Propagation

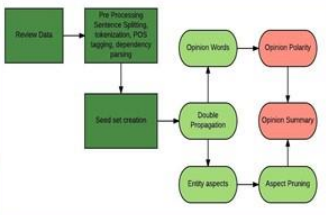
Ava Heinonen SID:1302575 arh144@student.anglia.ac.uk BEng Computer Science
Supervisor: Cristina Luca 03/2017



Anglia Ruskin University

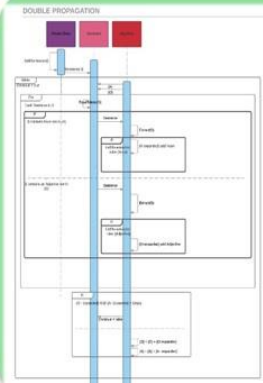
Aims and Objectives:
The aim of the project is to research unsupervised aspect extraction methods for aspect level sentiment analysis, and implement and improve upon an aspect extraction algorithm.

Selected Algorithm:
After examining a few alternatives, Double Propagation was selected.



Double Propagation:
Double propagation is syntax based unsupervised aspect extraction algorithm, which extracts aspects and opinion words simultaneously. It works by iterating over data extracting nouns and adjectives based on their linguistic dependencies on one another.

Methodology:
Python and spaCy were used to implement the algorithm. Dependency tags and extraction rules were adapted to work with spaCy. The original algorithm required a predetermined seed set of adjectives.^[1] A method was developed to create the seed set automatically using frequency distribution of adjectives in the data.

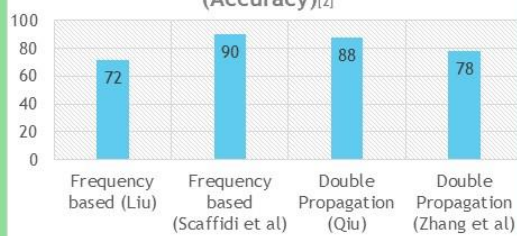


Rule: $Adj \rightarrow dep1 \rightarrow word \leftarrow dep2 \leftarrow Noun$
s.t Adj in {O}, dep1 or dep2 in {REL}

Long POS == ADJ and the title POS == NOUN long in {O} and nsubj in {REL}

[A-extended] add the title

Unsupervised Extraction Algorithms (Accuracy)^[2]



Algorithm	Accuracy (%)
Frequency based (Liu)	72
Frequency based (Scaffidi et al)	90
Double Propagation (Qiu)	88
Double Propagation (Zhang et al)	78

Testing:
The same pre tagged set of amazon review data as previously used by Qiu et al and Liu is used for the testing.

Results are compared against previous implementations of the algorithm (Qiu et al and Zhang) and other unsupervised algorithms.

[1] Chen, C., Bu, J., Liu, B. and Qiu, G. (2009). Opinion Word Expansion and Target Extraction through Double Propagation. *Proc. 21st Int. Joint Conf. Artif. Intell.*, pp.1199-1204.

[2] Frasnar, F. and Schouten, K. (2015). Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(3), pp.813 - 830.

2. Interim Report

Final Project Interim Report

MOD002691

*NOUN FREQUENCY BASED ASPECT EXTRACTION METHODS FOR
ASPECT LEVEL SENTIMENT ANALYSIS*

SID:1302575/2

Date:10/11/2016

Table of contents

1.0 PROJECT PROPOSAL FORM.....	3
2.0 PROJECT TIMELINE.....	4
2.1 PROJECT TIMELINE	
2.2 GANTT CHART	
3.0 ETHICS PASS CERTIFICATE.....	5
4.0 CV.....	6
5.0 EXIT PLAN	8

PROJECT PROPOSAL FORM

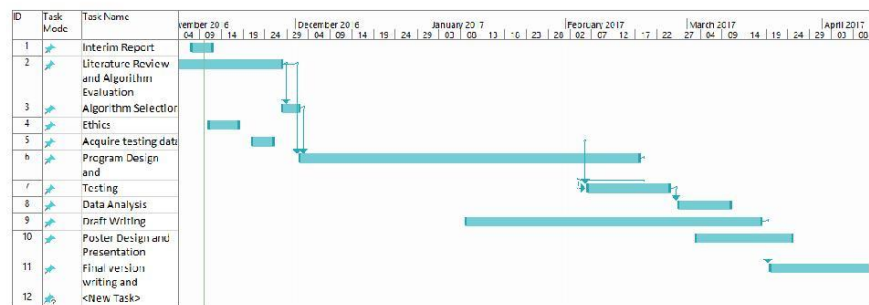
NAME:	Ava Heinonen
SID:	1302575/2
Email:	.ava.heinonen.@student.anglia.ac.uk
Degree course: <i>Include BSc/BEng designation</i>	BEng Computer Science
Start of project: month/year	October 2016
Expected project completion/ submission: month/year	April 2016
Draft Project Title:	Noun Frequency based aspect extraction algorithms for aspect level sentiment analysis
Possible supervisor: Suggest a supervisor or leave blank	Cristina Luca
Aim(s): Suggest one or two aims	Research, implement and test frequency based aspect extraction algorithm for aspect level sentiment analysis of product review data
Previous work: Give <u>two</u> literature sources relevant to this work you have consulted (optional, but will help verify the topic is worthy of study)	M. Hu, B.Liu.,2004. "Mining Opinion Features in Customer Reviews." in Proceedings of the 19th National Conference on Artificial Intelligence (AAAI 2004). AAAI, 2004, pp. 755–760. C.Scaffidi et al.,2007."Red Opal: Product-Feature Scoring from Reviews" in Proceedings of the 8th ACM Conference on Electronic Commerce (EC 2007). ACM, 2007, pp. 182–191.
METHODOLOGY AND OUTCOMES: Describe how you hope to achieve your aims and how you will measure the success of the work (100 words max)	-Literature based review and evaluation of currently used frequency based aspect extraction methods. -Implementation of an algorithm -Comparing the results produced by the algorithm to human tagged version of the same data and results achieved in previous implementations of the same algorithm.
Resources: eg software, hardware, industrial, human	-Python 2.7 -Python NLTK toolkit -Pre tagged product review data

2.0 PROJECT TIMELINE

2.1 PROJECT TIMELINE

Task Name	Duration	Start	Finish	Predecessors
Interim Report	5 days	Mon 07/11/16	Fri 11/11/16	
Literature Review and Algorithm Evaluation	20 days	Tue 01/11/16	Sun 27/11/16	
Algorithm Selection	4 days	Mon 28/11/16	Thu 01/12/16	2
Ethics	5 days	Fri 11/11/16	Thu 17/11/16	
Acquire testing data	5 days	Mon 21/11/16	Fri 25/11/16	
Program Design and Implementation	56 days	Fri 02/12/16	Fri 17/02/17	2,3
Testing	15 days	Mon 06/02/17	Fri 24/02/17	6,5
Data Analysis	10 days	Mon 27/02/17	Fri 10/03/17	7
Draft Writing	50 days	Mon 09/01/17	Fri 17/03/17	
Poster Design and Presentation	16 days	Fri 03/03/17	Fri 24/03/17	
Final version writing and handing in	30 days	Mon 20/03/17	Fri 28/04/17	9

2.1 GANTT CHART



3.0 ETHICS PASS CERTIFICATE

Word For... Online Photo ... Ethics - A... New Tab Question... X MAPAC | ... PODS | W... START ES... BT - Una... Anglia Ru...

https://myqmp.anglia.ac.uk/perception5/open.php?customerid=Perception

anglia.ruskin.harvard

Most Visited Getting Started Usable Privacy

Research Ethics Quiz for Health Sciences etc

Assessment Feedback

Congratulations, SID1302575/2

You scored 9 out of 10 (90 per cent) at 16:03 on Tuesday, 03 May, 2016

You have **PASSED** the Research Ethics Quiz.

Please take a screen capture of this information to attach to your Ethics Approval Application for submission to the relevant Research Ethics Panel.

You can also review how well you did on individual questions by scrolling down to view any feedback.

Total score: 9 out of 10, 90%

Question Feedback

1 of 10

1. Which of the following is most likely to raise a significant ethical issue? (select one).

- ☒ a) A senior manager in a mental health charity is doing a postgraduate degree and asks his junior staff to complete a questionnaire on workplace wellbeing.
- ☐ b) An Undergraduate hands out questionnaires about student experience and well-being as students enter the university with a box at reception for their answers.
- ☐ c) A Biology student is researching which flowers Victorian and Edwardian botanists chose to bring back to the UK.

1 out of 1

Ava Heinonen

Room 69, Study Inn Cambridge
Castle Court, Castle Street
CB3 0AU
Cambridge

077 766 067 834
avar.heinonen@gmail.com

OBJECTIVE	Motivated Computer Science student looking for an internship. Committed to postgraduate study in Machine Learning and Artificial Intelligence, and working towards a student dissertation research project in Sentiment Analysis.	
EDUCATION	ANGLIA RUSKIN UNIVERSITY, CAMBRIDGE BEng Computer Science, Expected graduation 2017 2:1 Anglia Ruskin University, Cambridge BA(hons) Film Studies and Media Studies, 2013-2014 Joensuu Niiinivaaran Lukio, Joensuu/Finland Finnish Matriculation Examinations (A-level equivalent), 2007 -2011 English, Mathematics, Biology and Finnish Prince of Songkla University International High School, Hat Yai/Thailand Rotary International student exchange program, 2009-2010	
RELEVANT COURSEWORK	Patient Monitoring: Object Oriented C# program simulating patient monitoring in a hospital setting. Software Design with UML. Automated testing and use of source control required. Project included training in Agile software development methods.	Sentiment Analysis: Third year major project in aspect level sentiment analysis using noun frequency based aspect extraction algorithms.
SKILLS	Programming Languages: c#, c++, Python, Matlab Operating Systems: Microsoft Windows, Linux Design: UML and ERD diagrams Other: SQL, git Networking: Completed two modules of Cisco accredited networking training Programs: Microsoft Visual Studio, Microsoft Visio, Python , Matlab, Languages: Finnish – Native English – fluent (IELTS 7.5) German – fluent Swedish – good	

**WORK
EXPERIENCE**

Anglia Ruskin Student's Union, Cambridge, UK

Administration Assistant, Sept 2016 – Current

Customer Service and Reception duties at the Student's Union office.

Database management.

Processing room bookings and speaker requests, and referring students to other ARU services as required.

Processing cash and card transactions.

Global Sustainability Institute, Anglia Ruskin, Cambridge, UK

Research Intern, March 2016 – May 2016

Conducted a pilot study on the feasibility of NUS Responsible Futures accreditation scheme Kite Mark in Anglia Ruskin University. This involved deciding the scope and the method of the study, collecting data and presenting it in a report sent to NUS and university staff. Assisted in data collection and analysis for two university wide studies for NUS responsible futures

Mumford Theatre, Cambridge, UK

Junior technical, Nov 2014 – Current

Building stage lighting and set structures for theater performances.

Anglia Ruskin Residential Services, Cambridge, UK

General Assistant, June 2015 – Sept 2015

Repairing and Maintaining university halls over the summer. Responsible for logistics of the freshers moving in period. Includes shifts as residential assistant responsible for responding to resident queries and emergencies overnight.

Card Factory, Cambridge, UK

Customer Assistant, Nov 2013 – May 2015

Customer service, till work, maintaining stock levels and product displays on the shop floor.

Joensuu Kaupungin Kirjasto, Joensuu, Finland

Librarian Intern, June 2013 – August 2013

Customer Service, organizing books and handling returned and reserved items.

Family Ursin, Bad Mergentheim, Germany

Au Pair, May 2012- May 2013

Worked as a live in nanny for a German family. Responsible for the daily care of two young children.

**HONORS &
ACTIVITIES**

Technical Manager, Cue.5 Anglia Ruskin Theater Society (May 2016 – Current)

Member of the elected board of the society.

Responsible for organizing and running workshops for students, and overseeing the technical team of all society productions.

WWOOF Volunteer, Germany (July 2014)

Worked together with small team of international volunteers to run organic farm/seminar center in Germany.

Rotary International student exchange program, Thailand (2009-2010)

Was chosen to represent my sending Rotary club and my country as an exchange student in Thailand.

EXIT PLAN

Ultimately my plan is to complete a doctorate degree and proceed into career in research and teaching in a good university.

To achieve this I have started to look into universities in Scandinavia for my master's degree.

Being a Finnish and EU national and speaking Finnish in a native level and good B1 Swedish, the universities in Finland and Sweden are open for me. Due to free, high quality universities back home I am aiming to study in Helsinki, Stockholm or Gothenburg.

University of Helsinki offers specialization in Algorithms, Data Analytic and Machine Learning, and university of Gothenburg has good selection of courses in Algorithms and Machine Learning, so I think either of them would be a good fit for me.

Both universities are also doing well on the world university rankings, so I trust I would have access to good doctoral programs with a master's degree from one of them.

University applications in Sweden are now open, so I will start working on my applications shortly. Applications for Finnish universities open in March, so I will work through another pile of applications then.

In the short term I will be finishing my studies concentrating on getting the grades required to proceed in my studies, and staying in the UK over the summer after I finish university.

I will be applying for summer internships now and during the spring semester to get some work experience in my own field.

I am also able to continue in my both current jobs over the summer if needed.

I also want to use the summer to volunteer in amateur theater, possibly again at the Edinburgh Fringe, as a technician, as I have done during the course of my studies.