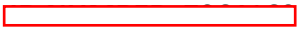
SCHOOL OF COMPUTER SCIENCE

THE UNIVERSITY OF MANCHESTER

AUTOMATIC TERM RECOGNITION

PROJECT BACKGROUND REPORT

STUDENT: DEMETRIS HADJITOFALLIS



SUPERVISOR: PROF. SOPHIA ANANIADOU

MAY 2010

Contents

List of Figures

Abstract

This report presents the results of the initial research and study has been done for the project. It contains a discussion about the wider context of the project and the related work to the subject studied, as well as analysis of the specifications of the tool which will be constructed during the project.

In particular, a brief introduction about 'Text mining' is provided, providing information about the history of the subject area, examples of applications that use Text Mining and a list of related subject areas. Then, the 'Automatic Term Recognition' (ATR) field is discussed, presenting some techniques used for ATR and focusing on the C/NC-Value (hybrid) method. Furthermore, the report contains a detailed presentation of the aims of the program and analysis of the method that will be used to develop it. In addition, the criteria of success are identified among with the evaluation methods will be used to test the program. Finally, the project plan is presented showing the deliverables and the deadlines, giving a clear view of all initial estimations.

Chapter 1

Introduction

1.1 Project motivation

The growing body of literature is becoming one of the most important problems that researchers face in any scientific domain, particularly in the biomedical one. The existence of an enormous number of scientific papers in a specific field makes the process of reading them very time-consuming, extremely 'painful' and even impossible for readers. The consequence of this is for the researchers to be restricted from discovering and associating information, and establishing hypotheses. Text mining offers a solution to this problem by suggesting approaches aiming at automatically discovering hidden and previously unknown information within documents.

"Terms are the linguistic representation of concepts and they are extremely important for digital libraries" [1]. The automatic identification and extraction of technical terms from literature it is extremely essential in order to completely understand a (scientific) document. As the body of literature keep growing, the result is the change of existing terms and the appearance of the phenomenon of neologisms ("from Greek νέος (neos 'new') + λόγος (logos 'word')" [2]) as the main characteristic of their terminologies. The need of methodologies targeting at acquiring and integrating information effectively and efficiently leaded to the evolution of a field, called Terminology Management. Its objectives are the terms identification from documents and their mapping to the appropriate concepts, in order to get access to the information that is stored in the literature.

This project focuses on the identification process, rather than in the mapping process. More precisely, it examines both the existing Automatic Term Recognition (ATR) approaches and the barriers to successful term identification, such as term ambiguity and term variation. [3] ATR systems are widely applicable in our days especially in Information Retrieval (IR) and Ontology Building applications. The great interest from the academic and scientific communities in this field leaded to the suggestion of many approaches and the development of several methods for the construction of an efficient and reliable ATR system. For the purpose of this project a method called 'C/NC-value' will be used which is based on a hybrid approach. This method is separated in two parts, the C-value part and the NC-value part; only the first part will be used in this project. 'C-value' combines statistical and linguistic information and it is mainly used for automatic recognition of (nested) multi-word terms, as well as for collocation extraction.

Chapter 2

Background

2.1 Relevant Literature

Text mining is the process of extracting high quality information (unstructured knowledge) that is hidden in the text and presenting it to the users in a simple way [4,5]. Text mining systems can be applied to natural language documents of any domain, identifying the concepts in the documents and providing previously unknown information, such as patterns and relationships. [6]. This process is very significant as it allows researchers to generate hypothesis based on those patterns and relationships.

Text mining was first introduced in the mid-1980s [4] but its significant evolution was made during the last decade; where the manual approaches where replaced by automatic approaches. The fact that almost 80% of the information is stored as text [7], was one of the major factors that increased the importance and the necessity of Text mining. Two additional reasons leaded to the increase of the commercial value of this field were the 'information overload' and the decrease of the level of satisfaction of the use of Information Retrieval (IR) techniques. [8]

A wide range of applications such as security, biomedical software online media, academic and marketing applications use text mining. However, the most popular applications are the biomedical applications. The reasons are that biomedicine is a dynamic and rapidly evolving field, and the knowledge of this domain is only in research papers. Thus, researchers of this domain are somehow obliged to read a huge amount of papers, where a majority of them might not be of particular interest.

"Typical Text mining tasks include text categorization, text clustering, concept/entity extraction, production of granular taxonomies, sentiment analysis, document summarization, and entity relation modelling (i.e., learning relations between named entities)." [4] Text mining pipeline is formed using techniques from a large pool of other subject areas such as Information Retrieval (IR), Natural Language Processing (NLP), Information Extraction (IE) and Data mining.

"The primary goal of an IR system is to retrieve all the documents which are relevant to a user query while retrieving as few non-relevant documents as possible". [9] The analysis of document collections, in text mining, is done with complex and severe algorithms. The use of IR systems, allow the decrease of the number of documents in a collection which leads to the increase of processing speed. On the other hand, the goal of NLP is to make computers understand the human language, constructing a computational model of language. This is achieved by analysis of the human language. This analysis might include part-of-speech tagging, word sense disambiguation and parsing. The

existing, different, types of analysis provide linguistic data to the text mining systems, usually by annotating documents, during the Information Extraction (IE) phase.

"IE is the process of automatically extract structured information, i.e. categorized, contextually and semantically well-defined data from a certain domain, from unstructured machine-readable documents" [10]. IE systems can carry out term analysis, named-entity recognition and fact extraction. The term analysis task identifies multi-word terms in a document. "Named-entity recognition seeks to locate and classify atomic elements in text into predefined categories such as the names of persons, organizations or locations" [11]. Finally, Fact Extraction aims at identifying facts such e.g. relationships. The data generated during IE is stored in a database. Data mining techniques are then applied in order to identify and extract patterns from data. [12] Data mining allows the discovery of hidden knowledge in a document, for example by exploiting the relationships and interactions between facts and identifying patterns. [6]

Two of the main organizations that focus on the Text mining field are the *'Tsujii Laboratory'* and the '*National Centre of Text Mining*' (NaCTeM). Tsujii Laboratory is a research group that focuses on Natural Language Processing (NLP) and Computational Linguistics (CL). [13]"CL is an interdisciplinary field dealing with the statistical and/or rule-based modelling of natural language from a computational perspective". [14] Alternatively, NaCTeM provides services and tools to the UK academic community, in cooperation of the University of Manchester and the University of Tokyo. [15] NaCTeM is building a wide range of resources and tools such as tokenisers, taggers, chunkers, parsers, NE recognisers, semantic analysers. It also provides a large number of services. [8] This project will use two tools produced by the NaCTeM, the '*GENIA Tagger*' and the *'TerMine'*, that will be explained later in the report.

2.2 Analysis of related work

2.2.1 Terms and Terminology

Every subject area consists of concepts, which are comprised by their underlying relationships. These concepts are encapsulated in the terms in the literature of the domains. Thus, the differentiation between terms and words is an essential and major task. "Terms are the linguistic representation of the concepts in a particular subject field, and are characterised by special reference; "aiming at classifying special knowledge" [5]. On the other hand, according to Sager et al, words function in general reference over a variety of codes. [16] So, while the words collectively form the vocabulary, the terms of a domain form its terminology. An additional definition states that "a term *T* can be defined as an ordered pair *<c, t>*, where '*c*' is a concept from a special language and '*t*' is a termform." [17] Terms are separated in two categories: simple and complex. Simple terms consist of only one word and complex terms consist of two or more words (also called multi-word terms).

CHAPTER 2: BACKGROUND

Terminology has a twofold meaning:

1. "the scientific field pertaining to the study of relations between concepts and their designations (terms, names and symbols) and the formulation of principles and methods governing these relations in any given subject field; and the task of collecting, processing, managing and presenting terminological data in one or more languages" [18]

2. "the set of terms belonging to the special language of an individual subject field" [18]

In order to introduce a new term in a subject, a concept that point to the specific term must exist. To classify a term we need to associate and group a concept with the other concepts in the domain, as a result of the creation of a new term for that concept. "Terms are not themselves knowledge items, but they refer to the concepts that are the knowledge items. Naming involves the use of domain-specific term formation patterns to label a concept introduced by a definition." [5,19] The creation of term formation patterns can be achieved in several ways:

1. Using existing linguistic or terminological resources that change the meaning of an existing word-form. This is achieved with the use of similes, metaphors, explores polysemy and homonymy in language. The drawback of this process is the cause of ambiguity. [5]

   Examples: [18]
   - **Circuit:**
     General Language -> a line enclosing a surface
     Electrotechnology -> an arrangement of devices or media through which electric current can flow
   - **Storage:**
     General Language -> putting objects or materials in a storage area
     Computer Science -> recording an active document/file on a magnetic or other means

2. Modifying existing term resources using transformations, such as affixation, compounding, abbreviations, etc. [5]

   Examples: [18]
   - **Compounding:** paraplegic + Olympics = Paralympics
   - **Abbreviations:** et cetera = etc.

3. Creating new linguistic entities (neologisms). Neologisms can be completely new inventions or may borrow heavily from other languages, typically Greek and Latin. Neologisms are often created by using some modification; e.g. a combination of words and numerals, letters, symbols, and ponyms. [5]

   Examples: [18]
   - Reservoir -> From French
   - Diameter -> From Ancient Greek

2.2.2 Term Ambiguity and Term Variation

A term may refer to many concepts and a concept may be referred by many terms, forming a many-to-many relationship. [5,20] The phenomena of *Term Ambiguity* and *Term Variation* are two major issues need to be addressed in the domain of term recognition.

Term Ambiguity refers to the situation where a term may refer to many concepts. It can be separated in syntactic and semantic ambiguity. [21] Syntactic ambiguity refers to the situation where a sentence can be parsed in more than one ways, opposed to semantic ambiguity that refers to the case where "a word or concept has an inherently diffuse meaning based on widespread or informal usage" [22]. The uses of Part-Of-Speech (POS) taggers, which can reach a level of accuracy between 95% -99%, decrease dramatically the probability of incorrect syntactic identification. Semantic ambiguity can be further divided in two categories; polysemy and homonymy. [21] Polysemy is the case where a term is related to more than one concept related in meaning, while homonymy is the case where two different terms have identical surface form and unrelated meaning.

Disambiguation methods based on machine learning approaches were introduced to deal with this phenomenon. These methods attempt to discover the right concept that is related with a term in a given domain. Many experimental methods appeared especially in the biological and biomedical field. For example, Hatzivasiliou et al. proposed techniques based, among other, on naïve Bayesian and decision trees. [23] On the other hand, Pakhomov experimented both with a technique used a maximum-entropy classifier, and with features based on document layout. [24] Finally, Liu et al. worked with a method that aimed to select a correct concept of a term and build a classifier for each sense of the term. [25]

As opposed to term ambiguity, *Term variation* is the phenomenon where different terms are associated with the same concept. This phenomenon is very usual as about one third of the occurred terms in a domain are variants; thus it is more intense on quickly growing domains/subject areas. Term variation has a significant effect on both automatic (performed by computers) and manual (perform by human experts) term recognition. [5] Term variation can be distinguished in six different types; Orthographic, Morphological, Lexical, Acronyms, structural (or syntactic) and semantic. Acronyms are the most important type of term variation as "they are used more frequently than full terms and there are no rules or exact patterns for the creation of acronyms from their full form" [21]. Figure shows some examples of variations.

Orthographic*:*	Memory-to-Memory Model **<=>** Memory to Memory Model B2B Model **<=>**b2b Model
Morphological*:*	Down's syndrome **<=>** Down syndrome
Lexical*:*	Cardiac disease **<=>** Heart disease
Structural*:*	SMRT and Trip-1 RNAs **<=>** SMRT RNA and Trip-1 RNA
Acronyms*:*	EBNF **<=>** Extended BNF **<=>** E-BNF **<=>** [E]BNF

Figure 2.2: Type of variations [5]

There are several methodologies which can be followed in order to deal with term variation. One of the approaches is based on stemming. “Stemming is the process for reducing inflected (or sometimes derived) words to their stem, base or root form.” [26] Two or more terms belong to the same equivalent class (i.e. a class that contains terms that represent the same concept) if they shared a stemmed representation. However, this method can lead to over-stemming or under-stemming. Another technique used to deal with this phenomenon “is the combination of terms and their synonyms from existing technologies with approximate string matching and edit distance techniques” [5]. Other techniques also exists are based, among other, on direct term synonyms identification or on variations rules. [5]

2.2.3 Terminology management

The application of ‘Text mining’ to different domains suffers from a number of severe limitations such as term variation, term ambiguity, existence of human-focused resources and other. The need of overcoming these limitations raises the necessity of the design of techniques for automatic terminology management. Terminology management is the process of formatting, storing and maintaining the term sets that belong to a distinct language of a particular domain (i.e. terminologies). Terminology management systems are distinguished in manual, semi-automatic and automatic. For the purpose of this project we will only consider automatic terminology management systems. Their major aim is the recognition of terms in texts and their association with the appropriate concepts. Such systems compose of three modules; an Automatic Term Recognition (ATR) module (this is what we are mainly interested in), an Automatic Term Structuring (ATS) module and an Intelligent Term Manager (ITM) module. ATR module identifies and extracts term lexical units from text while ATS module forms the relationships between terms by using classification and clustering. On the other hand, ITM module stores the terms in the appropriate repositories and provides information from external databases. [5]

The process of identifying terms in a document, or a collection of documents, typically consists of three tasks/steps (*Figure2.3*). The first task is ‘Term recognition’ in which we determine which words are associated with concepts, separating terms from non-terms. Then, these words are separated in different domain classes during the ‘Term classification’ task. Note that as terms expressing new concepts are automatically linked to particular parts of the ontology, this task, becomes extremely significant for ontology management [3]. Ontologies are conceptual models aiming at representing the structure of a domain and supporting the sharing and integration of knowledge by modeling (domain-specific or general) relations. [27] Finally, ‘Term mapping’ establishes term identity by mapping the terms into domain terminologies. To illustrate the term identification process, consider the following example: Let the sentence *“‘p53 protein suppresses mdm2 expression”* [3] included in a given document. The first task recognises ‘*p53 protein*’ and ‘*mdm2*’ as terms, while the second task classifies the ‘*p53 protein*’ as a protein and the ‘*mdm2*’ as a gene. The last step includes mapping of these two terms to reference data sources.

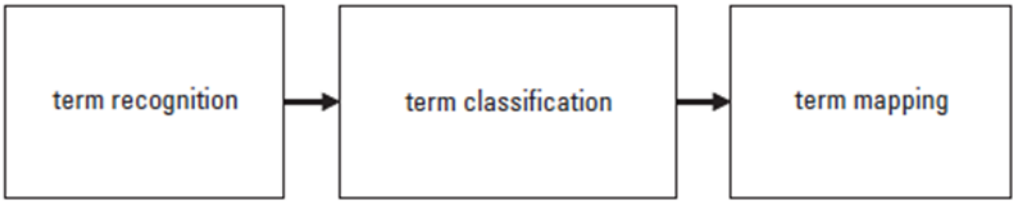


Figure 2.3: Steps in term identification in text [5]

2.2.4 Automatic Term Recognition

Automatic term recognition denotes the process that extracts of technical terms from special language corpora with the use of computers (i.e. huge and organized electronically-stored sets of text [28]) and representation of these terms to the user in a sensible manner [17,29]. ATR provides the ability to manipulate large volume of data, which will be impossible to be done manually. The fact that it provides the ability to handle unknown words, build lexical and knowledge resources and maintaining consistency avoiding terminological confusion, make it extremely significant for robust processing. [21] ATR is used in a wide range of applications such as indexing in digital libraries, indexing in books, text categorization, human and machine translation, specialised dictionary construction and hypertext linking. [17] ATR techniques are based on five different approaches; dictionary-based, rule-based, statistical, machine learning, and hybrid (combination of the previous) approaches.

The first approach assumes the existence of a number of terminological resources, i.e. lexical resources - dictionaries. A term is defined as the word (string), or a sequence of words, that matches an entry in one of the existing terminological resources. The problem with this approach is the decrease of the sensitivity of the ATR system, as a lot of terms may not be recognised as a result of the existence of neologisms and variations. [5] Experiments made by Hirschman, Morgan and Yeh [30] using a collection of gene names form 'FlyBase' database, proved the existence of these problems. The results for full articles, presented a low percentage of precision (2%) and a high percentage of recall (84%). Additionally, the results for abstracts, presented precision 7% and recall 31%. A solution to the problems raised for this approach might be the use of additional processing in conjunction with the use of terminological resources. Tsuruoka [31,32] proposed a probabilistic generator of spelling variants based on 'edit distance' operations aiming at increasing the IR recall and dealing with the problem of term variation. 'Edit distance' is considered as a threshold for a term to be characterised as spelling variant. [3]

Rule-based approach focuses on the construction of rules that describe different common naming structures for particular term classes using either simple (orthographic or lexical clues) or complex morpho-syntactic features. A term is defined as the word (string), or a sequence of words, that can be defined by such a rule. Note that dictionaries of typical term components, such as acronyms and affixes, might be used in order to support the term recognition process. [3] "A grammar-based approach was introduced by S. Ananiadou [33] which suggested the description of term formation patterns using a four level ordered morphology. The system used a morphological unification

grammar and a lexicon with instances of specific affixes, roots and Greek/Latin neoclassical combining forms." [3] The drawback of this approach has to do with the domain-specific nature of the created rules, which make both the tuning with different domain or classes problematic and the approach time-consuming. The use of Machine Learning and Statistical approaches aim to overcome this problem. [5]

The major objective of Machine Learning (ML) is the allowance of computers to evolve behaviours based on empirical data. [34] "ML systems are designed for a specific class of entities. They use training data to learn features that are useful and relevant for term recognition and classification." [5] A term is defined as the word (string), or a sequence of words, which satisfies fitness criteria (based on learned features). An example of a ML method is the one suggested by Collier, Nobata and Tsujii [35], proposing the use of both orthographic features and Hidden Markov Models. In this method candidate terms are classified according to orthographic similarity. [3] The three main issues that this approach needs to address are the selection of appropriate features, in order to achieve as much accuracy as possible, the identification of term boundaries of multi-word terms, and the finding of reliable training resources. [5]

"Statistical approaches are based on statistical distributions of collocations in text." [5] Collocation refers to the phenomenon of the co-occurrence of a sequence of words in a text, more frequently than would be predicted, unintentionally. [36] Statistical approaches have an advantage in contrast with the Machine Learning and Rule-based approaches as it overcomes the problems appeared in those approaches. However, they suffer from lack of existence of satisfactory measures of termhood (i.e. "the degree that a candidate term is related to a domain-specific concept" [37]) of identified terms. A potential solution to this problem is the extraction of noun phrases as term candidates, a huge percentage of terms are noun phrases, and then calculate the termhood of these candidates. Thus, a term is defined as the word (string), or a sequence of words, that is above a given threshold; this threshold is based on the occurrence of the candidate terms. [5]

Finally, Hybrid approaches use a combination of different approaches in order to make use of all the advantages each one offers, aiming to construct a more efficient ATR system. The major hybrid method, that will be presented in the report, examined and used during the project, is the *C/NC-Value*. Other hybrid methods are those introduced by Proux et al [38], which is based on a morphological POS tagger, and Rindflesch et al [39].

The performance of an ATR system is measured in terms of precision and recall. Precision is defined as the number of terms extracted divided by the total number of candidate terms extracted. It is a measure of exactness of the lexical units that are proposed as terms. On the other hand, Recall is defined as the number of terms extracted divided by the total number of terms in the corpus, and is a measure of completeness. [40] The objective of ATR systems to achieve both high precision and recall is extremely difficult as these two values are reversely proportional (i.e. "high recall can be typically achieved at lower precision points, and vice-versa" [3]). Thus, the performance of ATR systems is measured by the *'F-measure'* value which can be considered as a harmonic mean of the two values (*Figure 2.4*).

$$Precision = \frac{\text{True Positives (TP)}}{\text{True Positives (TR)} + \text{False Positives (FP)}}$$

$$Recall = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + False\ Nagative\ (FN)}$$

$$F - measure = 2 * \frac{\text{precision } * \text{ recall}}{\text{precision} + \text{recall}}$$

Figure 2.4: Precision, Recall and F-measure

2.2.5 C/NC-value method

C/NC-value is a domain-independent method that combines linguistic and statistical information in order to extract multi-word terms from special language corpora. [1] It is separated in two parts, the *C-value* and the *NC-value*. The first part extracts terms using a statistical measure called 'frequency of occurrence'. C-value is also sensitive to nested terms, thus it improves the extraction of nested multi-word terms and it is used for collocation extraction. [41,42] Alternatively, NC-value aiming both to create a method for the extraction of term context words and to the incorporation of information from term context words to the extraction of terms. Essentially, it targets to the overall improvement of the extraction of multi-word terms. [1]

C-value takes as input a Spoken Language (SL) corpus and produces as output a list of candidate multi-word terms, ordered by their termhood (i.e. c-value). [1] It is divided into two parts, the Linguistic part and the Statistical part. The Linguistic part constructs the list of the candidate terms while the Statistical part assigns the termhood measure to each term in the list, i.e. ranks the terms in the list.

The linguistic part consists of three tasks/steps. The first task is the Part-Of-Speech (POS) tagging, also called grammatical tagging, which is the process of marking up the words in a text with a grammatical tag such as 'noun', 'verb', etc. [43] There are tools that perform this task, called *POS taggers*, which may achieve accuracy over 93%. [44] Note that before the tagging takes place in to each word in the corpus, the text (corpus) must be broken up into meaningful elements (words); this process is called tokenization. [45] Some taggers can also perform tokenization.

Tagging is required in order to be able to execute the second task which is the application of the linguistic filter. Linguistic filters aim to decrease the number of undesirable strings extracted. They distinguished in 'closed filters' which are very exact on what string they permit and in 'open filters' which are permit more types of strings. [1] The choice of the linguistic filter is extremely significant, as it has direct impact at the precision and recall of the output. The use of a 'closed filter' will improve precision but affect negatively the recall, while the use of an 'open filter' will improve the recall having a negative effect on the precision.

The final task of the linguistic part of the C-value method is the creation and application of the Stop-list. The objective of the Stop-list is similar as that of the Linguistic filter; it is used in order to permit the extraction of words that are not terms. The choice of what words (strings) we will include in the

Stop-list will have direct impact on the Precision and the Recall, as well. The use of such a list improves significantly the precision but we need to be very careful in order to not miss a (multi-word) term that includes a string from the Stop-list.

In contrast, the statistical part includes the calculation of the termhood for each term and the ranking of the list of candidate terms. The calculation is done taking into account the following four parameters:

1. The total frequency of occurrence of the candidate string in the corpus: This measure is used due to the fact that terms are usually used very frequently in a corpus. The equation for the calculation of the c-value for a term after considering the first parameter will be:

$$\boldsymbol{termhood}(\boldsymbol{a}) = \boldsymbol{f}(\boldsymbol{a})$$

$\boldsymbol{a}$ *is the candidate string,*
$\boldsymbol{f}(\boldsymbol{a})$ *its frequency of occurrence in the corpus*

2. The frequency of the candidate string as part of other longer candidate terms: In case there exists a multi-word term, its substrings will also be extracted as terms as they will occur at list as many times as the term. However, there might be the case that one (or more) of the substrings are not terms. In order to overcome this problem, we extract a substring as a term only if it appears a satisfactory number of times by itself in the corpus. [1] Thus, the equation will change to:

$$\boldsymbol{termhood}(\boldsymbol{a}) = \boldsymbol{f}(\boldsymbol{a}) + \sum_{\boldsymbol{b} \in \boldsymbol{Ta}} \boldsymbol{f}(\boldsymbol{b})$$

$\boldsymbol{a}$ *is the candidate string,*
$\boldsymbol{f}(\boldsymbol{a})$ *its frequency of occurrence in the corpus*
$\boldsymbol{Ta}$ *is the set of candidate terms that contain a*
$\boldsymbol{b}$ *is such a candidate term*

3. The number of these longer candidate terms: The solution given to the previous problem does not completely solve it. We define 'nested terms' as "those that appear within other longer terms, and may or may not appear by themselves in the corpus" [1]. The previous formula does not allow the extraction of terms that are only found nested, or if they have not found nested and they have not appear a satisfactory number of times by themselves. A more precise solution is to take into consideration the independence of a substring of a term. The inclusion of a substring (as nested term) in many longer terms a satisfactory number of times, indicates that that substring is definitely a term.

4. The number of words (length) contained in the candidate string: The last parameter donates that the probability of a shorter string to appear in the corpus 'Z' times is higher than the probability of a longer string to appear the same amount of times. Thus, the appearance of a longer string in the corpus is much more significant. The parameter "is based on the assumption that the probability of occurrence of a word in a corpus in independent from the probability of occurrence of any other word" [1,46].

Summarizing the parameters above and considering that some terms are not nested, we distinguish the calculation of termhood (c-value) in two cases. The first is the case where a string has not found nested and it is of maximum length. In this case the c-value will be calculated considering the length and the total frequency of the string. The second is the case where a string is not of maximum length and it is a substring of longer candidate terms. The fact that a string is part of longer candidate terms has negative effect in the c-value. However, this phenomenon can be moderated if the number of candidate terms that a string is part of is big. A big number of candidate terms, shows high independence of the string from these terms. [17]

$$\text{C-value(a)} = \begin{cases} log_2|a| \cdot f(a) & \textbf{\textit{a is not nested}} \\ log_2|a| \cdot \left(f(a) - \frac{1}{P(T_a)} \sum_{\boldsymbol{b \in Ta}} f(b)\right) & \textbf{\textit{otherwise}} \end{cases}$$

$\boldsymbol{a}$ *is the candidate string,*
$\boldsymbol{f}(\cdot)$ *is its frequency of occurrence in the corpus*
$\boldsymbol{Ta}$ *is the set of extracted candidate terms that contain a*
$\boldsymbol{P(Ta)}$ *is the number of these candidate terms*

The list with candidate terms produced as output from the execution of the C-value method is further refined during the execution of NC-value method. The NC-value method takes into account context information of candidate terms and incorporates them into the C-value method. Two steps must be followed in order to achieve the further refinement of the list of candidate terms.

The first step consists of two processes, the extraction of the term context words and the calculation of their weights. The extraction of term context words is done by selecting the candidate terms appeared at the top of the list produced by the C-value method. These terms appear to have high precision. Although we can experience some 'noise' with the way we implement this process, by finding some strings that are not terms, we accept it in order to achieve full automation. The calculation of the weight for each of the selected terms is done having as criterion "the number of terms it appears with" [1]. The equation used for the calculation of the weights of the terms is given below. The result of the equation can be though as the probability of a word 'w' to be a term context word. [1]

$$weight(w) = \frac{t(w)}{n}$$

$\boldsymbol{w}$ *is the context word to be assigned a weight as term context word*
$\boldsymbol{weight(w)}$ *is the assigned weight to the word w*
$\boldsymbol{t(w)}$ *the number of terms the word w appears with*
$\boldsymbol{n}$ *the total number of terms considered*

The second step of the NC-value includes re-ranking of the list produced as output from the C-value method, taking into account the contextual information. The aim of the re-ranking is to move the real terms closer to the top of the list. This process is taking place as follows: "Each term of that list is associated with a collection of context words in the corpus. If a word has been assigned a weight before then it keeps that weight, else a zero value will be given in its weight. A context factor is obtained for each candidate term by adding the weights for its term context words, multiplied by their frequency appearing with this candidate term". [1]

Chapter 3

Research Methods

3.1 Project Description

The project is divided into three main parts; the research, the programming and the writing of thesis. The objective of the first part is to investigate the Automatic Term recognition (ATR) field of the Text mining subject area. More precisely, it focuses on the familiarization on the existing approaches for the construction of *ATR* systems and on deeply understanding of how the hybrid method *C/NC-Value* works. The knowledge gained from the research undertaken will be used for the construction of a program that automatically extracts terms from a collection of documents which is the target of the second part of the project.

The research part of the project could be distinguished in two subparts. The first subpart can be considered as a brief introduction to the subject. It will consist of discovery of general information about the area and familiarization with the aims of Text mining as well as applications of Text mining systems. Oppositely, the second subpart will explain the major research of the project. It will focus on the investigation of techniques used for Automatic Term Recognition, emphasizing on a hybrid method called C/NC-value.

The programming part of the project involves design and development of a tool which will meet the need of applications, such as IR and Ontology Building applications, for automatically discovering the most important terms in a collection of documents. The tool will accept as input a collection of documents in the form of row text, i.e. ascii. The program will use the C-value method in order to produce as output a list of the most important terms in the documents. This list will consist of the candidate terms and a value associated with each term, called c-value, representing the termhood. The terms will be ranked according to that value.

The last part of the project will be the writing of the thesis. The thesis will present the research, the findings, the conclusions and the judgments produced during the project. It will also demonstrate a series of tests will be performed for the tool during the testing phase, aiming at ensuring its correct and smooth operation, as well as the results of its evaluation.

3.2 Criteria of success

3.2.1 Project evaluation

In order to characterise the project successful, its three parts must fully completed. The quantity of the sources, such as scientific articles, website and books, and the quality of the findings from those are the criteria of the success of the research part of the project. On the other hand, the correct design, the proper development and, most importantly, the proper functionality of the proposed tool, are the parameters which will define the success of the programming part of the project. The last part of the project, the writing of the thesis, must be done according to the regulation of the university. It will present and explain all the findings of the research, and explain in detail the steps taken during the design and the development of the program. Finally, the thesis will be the major means of evaluation of both the program, by providing evidence of its correct operation (see *Section 3.2.2*), and the whole project, as it will basically sum up everything done.

Both the main and the secondary objectives of the project were identified and clarified by my supervisor and me in an earlier state. However, they might be modified or changed accordingly along the way, if proven necessary. The main objectives are those that *must* be completed and include:

- Perform research on the Automatic Term Recognition (ATR) field
- Specifically focus, analyse and understand the hybrid method C/NC-value
- Construct a tool (program) that accepts as input a collection of documents, apply the C-value method and produced as output a list of the most important terms, ranked according to their termhood *(C-value)*; using properly a software engineering methodology

On the other hand, a secondary objective was set which *might* be completed if there is enough time left:

- Integrate the constructed tool to the U-Compare language processing system [47]

3.2.2 Program evaluation methods

As it was mentioned above, the tool that will be constructed will extract the most important terms from a collection of documents. The correct operation of the program will be determined according to the correctness of its results. There will be used two evaluation methods. The first evaluation method will be the comparison of the list of candidate terms for a given collection of documents produced by the tool, against the list of candidate terms produced manually for the same collection of documents by a domain expert. The use of a domain expert allows the contrast of the program's results with what would be the ideal results, as the domain expert will extract all the actual (real) terms from the corpus. The second evaluation method will be the comparison of the list of candidate terms for a given collection of documents produced by the tool, against the list of candidate terms

produced for the same collection of documents by the TerMine tool. TerMine also uses the C/NC-value method for automatically extract the most important terms in a collection of documents. Thus, the contrast of its results with the results of the created program will give the degree of 'relative' correctness; by 'relative' correctness I mean the degree of correctness of the program assuming that the results produced by the TerMine are the desirable (correct) results.

Note that the tool I will produce it will not deal with acronyms. Thus, there will be a direct impact on the results produced. Although it is extremely important for an ATR tool to be able to "understand" acronyms it is, in fact, very difficult to deal with them. Solutions to efficiently address these types of variations are outside of the scope of this project and will not be taken into consideration due to their complex nature and due to lack of time.

TerMine

TerMine is a service provided by the NaCTeM aiming at automatically identifying the most significant terms in a collection of documents and rank these terms according to their importance. [48] It uses the C-value hybrid method. A difference between this tool and the tool which will be created during the project is that TerMine is dealing with acronyms by proposing possible expansions for the identified acronyms. *Figure 3.1* shows the overall architecture of TerMine which has many similarities with the architecture of the tool we will construct. It can be used to build controlled vocabularies, collecting together words (single-word terms) or phrases (multi-word terms) of a particular subject area. [49]

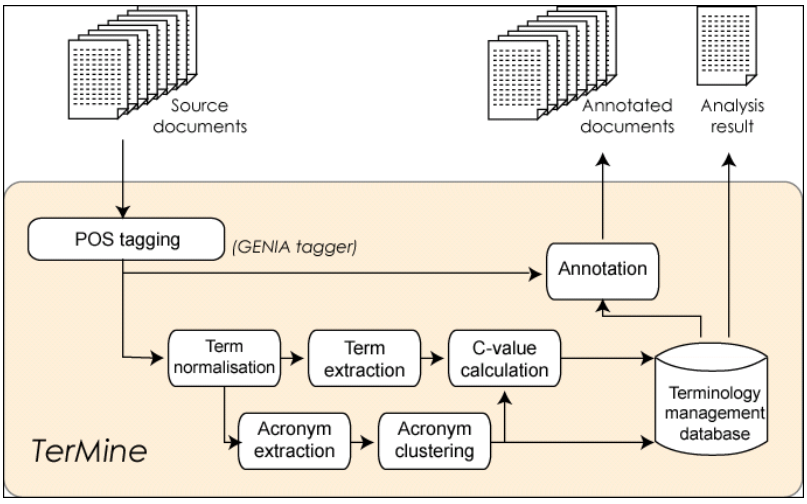


Figure 3.1: Overall architecture of TerMine [49]

3.3 Program specifications

3.3.1 Program algorithm

The tool will be constructed will implement the C-value method using the algorithm described in detailed in '*Appendix B'*. The output will be a list of candidate terms assigned with a value which will determine the importance of each term(C-value). Thus, the steps I am planning to follow in order to construct the tool are presented below:

- Input a collection of documents
- Apply the linguistic part of the C-value
    - Use Genia tagger to tag the corpus
    - Considering that the most terms and noun phrases, the linguistic filter will be used is: '*Noun+Noun* ***OR*** *(Adj|Noun)+Noun*'
- Apply the statistical part of the C-value
- Present the output to the user
    - All the intermediate results will be stored and presented to the user, if required

The development of the tool will be done according to the principles of separation of concerns and reuse of code. So, the functionality of the provision of the input, the processing of the documents (i.e. application of the algorithm) and the functionality of the presentation of results will be placed in different classes. Further separation between the linguistic and the statistical processing may be required. A significant amount of effort will be put in order to ensure the higher possible reuse of code in order to avoid complexity and achieve minimisation of classes (i.e. program lines).

Having those in mind, I have moved to the design of the first prototype for GUI of my program which is included in 'Appendix C'. The user will be able to be navigated from the Welcome Screen to the Input Screen by clicking the 'Enter' button. Then he can choose the desired documents using the file chooser at the left of the screen. The extraction process will start when the user will click the 'Extract' button. The results will be presented to the user on the Output Screen. At the right hand side area there will be presented the list of terms and on the left hand side area will be shown the provided document(s) with the recognised terms highlighted. The user can interact with the tool by choosing (from the appropriate Combo-Box) which document would like to see and which threshold of the C-value would like to apply to the list (table).

3.3.2 External tools choice

As it was mentioned before in the report, we are going to accept a level of noise for sake of automation of the term extraction process. However, we do care to decrease the noise as much as possible. In order to achieve this, we need to choose a Part-Of-Speech tagger which performs with as high precision as possible. A high precision degree is interpreted as a high degree of accuracy and a

high degree of performance. [50] For the purpose of program that will be constructed, I have chosen to use the GENIA tagger. "GENIA tagger is a general-purpose tagger that analyzes English sentences and outputs the base forms, part-of-speech tags, chunk tags, and named entity tags." [51] In addition, this tagger also has the ability to perform tokenization before the tagging process. The tokenization process is based on the 'Penn Treebank tokenization' technique [52] developed by the Computer and Information Science Department of the University of Pennsylvania. 'Penn Treebank tokenization' technique is characterized as "intelligent tokenization" due to the holding assumption that the sentences have been split before the start of the tokenization process. This enables the differentiation of the treatment of the end-of-sentence punctuation from the others, giving a context-sensitivity nature to the tokenizer. [53]

The performance of the tagger is illustrated in *Figure3.2* where a comparison of the performance of the tagger between Wall Street Journals (WSJ) and GENIA corpus is presented. A contrast of the performance of GENIA tagger with taggers trained on the WSJ corpus and on GENIA corpus is offered as well.

	Wall Street Journal	**GENIA corpus**
A tagger trained on the WSJ corpus	97.05%	85.19%
A tagger trained on the GENIA corpus	78.57%	98.49%
GENIA tagger	96.94%	98.26%

Figure 3.2: Tagging accuracies of different taggers [51]

3.3.3 Programming Language

There is a wide range of programming languages available to choose for the implementation of the program, such as C++, C#, Java, etc. Although the programming language choice will play a dominating role neither in the development nor in the success of the program, the choice of the proper programming language will be beneficial in a lot of ways. The main parameters that affected the selection of the programming language were the high level of knowledge of the selected language, as well as the ability of the language to create a nice-looking Graphical User Interface (GUI). The first criterion has to do both with the production of high quality work and with the decrease of the time needed to complete the program. The second criterion has to do with the ability of the language to create a good-looking GUI for the presentation of the results.

Taking into account these parameters, I decided that the use of Java as the programming language for the implementation of my program would be the most proper choice. The fact that I am particularly familiar with Java provides me the ability to produce high quality work quickly. The existence of Java libraries gives the capability to the programmer to overcome different issues raised during the programming phase easily. Additionally, instead of putting effort trying to learn a new programming language, which would be very time-consuming, I will focus on the actual challenges of the program; and generally of the project. Finally, I am also planning to use the NetBeans Interactive Development Environment (IDE), which is a Java IDE, which will simplify the construction of the GUI

of the program. This tool will save me time and effort in creation of the GUI, as its manual development it is not a straightforward task.

Despite the reasons of the selection of Java programming language have been presented above, Java has many other advantages that make it widely used. One of the main advantages of Java is that it is platform independent, moving easily from one computer system to another. [54] Furthermore, it is object-oriented language, providing the ability to the programmer to generate and handle objects, allowing the writing of reusable code. [54] However, the feature that makes Java widely usable is its simplicity; Java is a straightforward programming language allowing the programmer to write and compile a program easily, and understand a Java program without putting a lot of effort. The reason of its simplicity is the existence of automatic memory allocation and garbage collection. [54]

3.4 Organisation of work

The organisation of the work is the most significant aspect of a project. A good organisation could lead to the reduction of the time needed for each task or part and it might also have a direct impact to the success of the project.

3.4.1 Design and Development Methodology

The objectives of the program development phase were the production of an efficient and user-friendly Graphical User Interface (GUI), the writing of proper-style and reusable code, and the separation the presentation logic from the functional process logic. The first objective is essential so the program could present the results to the user in the appropriate form and the user will be able to interact with it effortlessly. The other objectives ensure the efficient modification of the program code (i.e. the programmer should be able to alter any functionality of the program efficiently) and the avoidance of producing complex code (i.e. the program code must be readable).

“In order to achieve these goals, I have chosen as basis for the development process I will use the Iterative and Evolutionary Methodology presented at the *Figure 3.3*. This methodology suggests the separation of the project’s work in a number of mini projects called ‘iterations’. Each iteration follows four phases; requirements analysis, design, implementation and testing. After the completion of each iteration, the outcome is integrated with the rest of the system so there is an incremental growth of the whole system. This allows the provision of useful feedbacks from the testing phases. Using this development methodology as a basis to organize my work will have a number of benefits. The progress will be visible in early stages and the complexity will be kept manageable. Also, based on this methodology rather than ‘Waterfall’, that is the most common software engineering methodology, I will gain better productivity, reducing the probability of project failure. More information about this process can be found in ‘Applying UML and Patterns’ book [55].” [56]

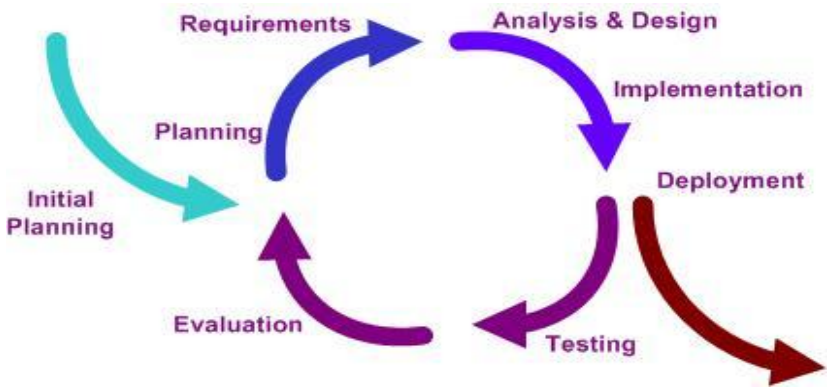


Figure 3.3: Iterative development [57]

3.4.2 Project Plan

I organized my work considering that the project was simple, small-to-medium size and created by only one person. I estimated that all main objectives will be completed, so I allocated my time having that in mind. In case that the allocated tasks will be finished early and enough time will be left, it will be allocated properly for the implementation of the secondary objective. The choice of the secondary objective has been done in collaboration with my supervisor.

Full details about the tasks planning of the project can be found at the '*Appendix A*' which includes a Gantt chart. The first part of the project is the research in the subject area. I separated this part in two subparts; the general research to become familiar with the wider context of the project and the specific research that includes research about the Automatic Term Recognition (ATR) emphasizing of the 'C/NC-Value' hybrid method. This task is estimated to take about nine (9) weeks, finishing early on April. An important load of the findings will be presented in the Initial Background Report which will be submitted on 14th of May. Thus, from the beginning of April until that date a significant amount of time will be spent in the writing of the report.

During this period, I will also start the programming part of the project, which will be separated in three sub-parts; the construction of the input interface, the construction of the main algorithm and the construction of the presentation GUI. The start of this part will start on the middle of April and it is estimated to finish early in June. Because of my choice of the Iterative methodology, time is allocated for the design, development and testing of each sub-part. After the construction of the program its evaluation will take place; a week it is allocated for comparing the results of the program with the results taken from TerMine and the results produced manually with the help of a scientific expert.

Finally, the time left, from the middle of June until 10th of September, is allocated for the writing and the correction of the thesis. Note that both the exam period and the time needed for studying the courses during the semester were taken into consideration during the construction of the project plan.

Conclusion

The growing of many scientific domains in conjunction with the need of accessing information stored in literature were the main reasons which leaded Automatic Term Recognition to become more than essential in our days. Despite the problems of term variation and term ambiguity that an ATR system may face, it has the ability, among other, to extract new terms of a particular domain which make it extremely useful. The main aim of the research triggered from this need, is the development of tools that automatically extract terms from the documents, discover their relationships and link them to the appropriate concepts. “Without knowledge of the terminology, the understanding of scientific documents is hindered.” [5]

This report firstly presents a brief overview of the wider context of project which is the Text mining subject area. Then, it presents the Terminology management process explaining the notion of ‘Terms’ and ‘Terminology’. Additionally, the problems of term variation and term ambiguity are explained. The final part of the first chapter focuses on the ATR field emphasizing on the C/NC-value method. Finally, the second chapter presents the program specification, the methodology to be followed for the design, the implementation and the testing of the program, as well as the overall organisation of the project.

This project aims at the exploration of the approaches taken for the development of an efficient ATR method; that are dictionary-based, rule-based, statistical, machine learning, and hybrid (combination of the previous) approach. The emphasis is given in the hybrid approach and particular in the C/NC-Value method. This method combines linguistic and statistical techniques which allow it to perform better than any other approach. In the meantime, a tool will be constructed in order to illustrate the advantages and the efficiency of the C-value method. This tool will accept a collection of documents, in the form of row text, as input and produced a list of the most important terms in this collection of documents, ranked according to their termhood (C-value).

Although a lot of progress has been made from the researches made during the recent years, there are still many challenges and issues out there for further research. Some of these challenges, as described in [5], are “the accurate recognition of term boundaries and further identification of internal term structure, the treatment of various types of term variation and their integration into term identification, the selection of the most representative terms (and concepts) in a document and the Anaphora resolution [58] and linking term coreferences” [5].

References

[1] K. Frantzi, S. Ananiadou, and H. Mima, "Automatic Recognition of Multi-Word Terms: the C-value / NC-value Method," *International Journal on Digital Libraries*, 1998.

[2] (2010) Wikipedia: Neologism. [Online]. http://en.wikipedia.org/wiki/Neologism

[3] M. Krauthammer and G. Nenadic, "Term Identification in the Biomedical Literature," *Journal of Biomedical Informatics*, vol. 37, no. 6, pp. 512-526, Dec. 2004.

[4] (2010) Wikipedia: Text mining. [Online]. http://en.wikipedia.org/wiki/Text_mining

[5] S. Ananiadou and J. McNaught, *Text Mining for Biology and Biomedicine*. Boston | London, United Kingdom: Artech House, Inc., 2006.

[6] J. Redfearn and NaCTeM members. (2008) Jisc Text Mining. [Online]. http://www.jisc.ac.uk/publications/briefingpapers/2008/bptextminingv2.aspx

[7] S. Grimes. (2010) Unstructured Data and the 80 Precent Rule. [Online]. http://www.clarabridge.com/default.aspx?tabid=137&ModuleID=635&ArticleID=551,

[8] S. Ananiadou. (2010) Text Mining for Biomedicine: Techniques & tools. [Online]. http://www.nactem.ac.uk/dtc/DTC-Ananiadou.pdf

[9] B. Ribeiro-Neto and R. Baeza-Yates, *Modern Information Retrieval*, 2nd ed. Addison-Wesley-Longman Publishing co., 1999, http://people.ischool.berkeley.edu/~hearst/irbook/1/node2.html#SECTION00111000000000000000.

[10] (2010) Wikipedia: Information extraction. [Online]. http://en.wikipedia.org/wiki/Information_extraction

[11] (2010) Wikipedia: Named-Entity recognition. [Online]. http://en.wikipedia.org/wiki/Named_entity_recognition

[12] (2010) Wikipedia: Data mining. [Online]. http://en.wikipedia.org/wiki/Data_mining

[13] (2010) Tsujii Laboratory. [Online]. http://www-tsujii.is.s.u-tokyo.ac.jp/

[14] (2010) Wikipedia: Computational Linguistics. [Online]. http://en.wikipedia.org/wiki/Computational_linguistics

[15] (2010) NaCTeM. [Online]. http://www.nactem.ac.uk/index.php

[16] J. C. Sager, D. Dungworth, and M. P. F., *English Special Languages*. Wiesbaden, Germany: Oscar Brandstetter Verlag KG, 1980.

[17] K. T. Frantzi and S. Ananiadou, "Automatic Term Recognition using Contextual Cues," Dept. of Computing, Manchester Metropolitan University, 1997.

[18] K. Valeontis and E. Mantzari, "The linguistic dimension of terminology: Principles and methods of term formation," ELETO (Hellenic Society for Terminology), 2006.

[19] J. C. Sager, "Term Formation," *Handbook of Terminology Management*, vol. 1: Basic Concepts of Terminology Management, pp. 25-41, 1997.

[20] I. Meyer, K. Eck, and D. Skuce, "Systematic Concept Analysis Within a Knowledge-Based Approach to Terminology," *Handbook of Terminology Management*, vol. 1: Basic Concepts of Terminology Management, p. 98–118, 1997.

[21] S. Ananiadou. (2010) Lecture 10a: Automatic Term Recognition. COMP30421: Natural Language Engineering.

[22] (2010) Wikipedia: Ambiguity. [Online]. http://en.wikipedia.org/wiki/Ambiguity

[23] V. Hatzivassiloglou, P. A. Duboue, and A. Rzhetsky, "Disambiguating Proteins, Genes,and RNA in Text: A Machine Language Approach," *Bioinformatics*, vol. 17, no. 1, p. 97–106, 2001.

[24] S. Pakhomov, "Semi-Supervised Maximum Entropy Based Approach to Acronym and Abbreviation Normalization in Medical Texts," in *Proc. 40th ACL Conference*, 2002, pp. 160-167.

[25] H. Liu, S. B. Johnson, and C. Friedman, "Automatic Resolution of Ambiguous Terms Based on Machine Learning and Conceptual Relations in the UMLS," *J. Am. Med. Inform. Assoc*, vol. 9, No 6, pp. 621-636, 2002.

[26] (2010) Wikipedia: Stemming. [Online]. http://en.wikipedia.org/wiki/Stemming

[27] I. Spasic, S. Ananiadou, J. McNaught, and A. Kumar, "Text mining and ontologies in biomedicine: Making sense of raw text," *Brief Bioinform*, vol. 6, no. 3, pp. 239-251, Jan. 2005.

[28] (2010) Wikipedia: Text corpus. [Online]. http://en.wikipedia.org/wiki/Text_corpus

[29] I. Mani and M. T. Maybury, "MIT Press: Advances in Automatic Text," Cambridge, Massachusetts, USA, 2003.

[30] L. Hirschman, A. A. Morgan, and A. S. Yeh, "Rutabaga by any other name: extracting biological names," *J Biomed Inform*, vol. 35, no. 4, pp. 247-259, 2002.

[31] Y. Tsuruoka and J. Tsujii, "Probabilistic Term Variant Generator for Biomedical Terms," in *26th Annual ACM SIGIR Conference*, 2003, pp. 167-173.

[32] Y. Tsuruoka and J. Tsujii, "Improving the Performance of Dictionary-Based Approaches in Protein Name Recognition," *Journal of Biomedical Informatics, Special Issue on Named Entity Recognition in Biomedicine*, vol. 37, No. 6, pp. 461-470, 2004.

[33] S. Ananiadou, "A Methodology for Automatic Term Recognition. .," in *COLING-94*, Kyoto, Japan, 1994, pp. 1034-1038.

[34] (2010) Machine Learning. [Online]. http://en.wikipedia.org/wiki/Machine_learning

[35] N. Collier, C. Nobata, and J. Tsujii, "Extracting the Names of Genes and Gene Products with a Hidden Markov Model," in *COLING 2000*, Saarbruecken, 2000, pp. 201-207.

[36] (2010) Wikipedia: Collocation. [Online]. http://en.wikipedia.org/wiki/Collocation

[37] I. Korkontzelos, I. P. Klapaftis, and S. Manandhar, "Reviewing and Evaluating Automatic Term Recognition Techniques," Department of Computer Science, The University of York.

[38] D. Proux, F. Rechenmann, L. Julliard, V. V. Pillet, and B. Jacq., "Detecting Gene Symbols and Names in Biological Texts: A First Step toward Pertinent Information Extraction," in *Ninth Workshop on Genome Informatics*, 1998, pp. 72-80.

[39] T. C. Rindflesch, L. Hunter, and A. R. Aronson, "Mining molecular binding terminology from biomedical text," in *AMIA Symp.*, 1999., pp. 127-131.

[40] (2010) Wikipedia: Precision and Recall. [Online]. http://en.wikipedia.org/wiki/Precision_(28information_retrieval)

[41] K. T. Frantzi, S. Ananiadou, and J. Tsujii, "Extracting terminological expressions," *The Special Interest Group Notes of Information Processing Society of Japan*, vol. 96-NL-112, pp. 83-88, 1996.

[42] K. T. Frantzi and S. Ananiadou, "Extracting nested collocations," *Proceedings of the 16th International Conference on Computational Linguistics*, vol. COLLING'96, pp. 41-46, 1996.

[43] (2010) Wikipedia: Part-Of-Speach tagging. [Online]. http://en.wikipedia.org/wiki/Part-of-speech_tagging

[44] A. Gelbukh, "Computational linguistics and intelligent text processing," in *4th International Conference - CICLing 2003*, Mexico City - Mexico, 2003, p. 158.

[45] (2010) Tokenization. [Online]. http://en.wikipedia.org/wiki/Tokenization

[46] T. Dunning, "Accurate methods for the statistics of surprise and coincidence," *Computational Linguistics*, no. 19 (1), pp. 61-74, 1993.

[47] (2010) U-Compare. [Online]. http://u-compare.org/index.html

[48] S. Ananiadou and JISC Communications team, "National Centre for Text Mining: An introduction to tools for researchers," Sep. 2008.

[49] S. Ananiadou, "The National Centre for Text Mining: A Vision for the Future," *Ariadne* , no. 53, p. TheNationalCentreforTextMining:AVisionfortheFuture, Oct. 2007.

[50] Y. Tsuruoka, et al., "Developing a Robust Part-of-Speech tagger for Biomedical Text: Advances in Informatics," in *10th Panhellenic Conference on Informatics*, 2005.

[51] Y. Tsuruoka. (2010) GENIA tagger. [Online]. http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/

[52] (2010) Penn Treebank Tokenization. [Online]. http://www.cis.upenn.edu/~treebank/tokenization.html

[53] (2008) LingPipe Blog: Natural LAnguage and Text Analysis. [Online]. http://lingpipe-blog.com/2008/06/26/the-curse-of-intelligent-tokenization/

[54] S. Munaf. (2007, Apr.) Java Advantages and Disadvantages. [Online]. http://www.webdotdev.com/nvd/content/view/1042/204/

[55] C. Larman, *Applying UML and Patterns*, 3rd ed. Upper Saddle River, USA: Prentice Hall PTR, 2005.

[56] D. Hadjitofallis, "A Grammar Wizard," School of Computer Science, The University of Manchester, Manchester, Third Year Project Report, 2009.

[57] (2010) Wikipedia: Iterativedevelopment. [Online]. http://en.wikipedia.org/wiki/Iterative_development

[58] X. Yang, J. Su, G. Zhou, and C. L. Tan, "Improving Pronoun Resolution by Incorporating Coreferential Information," in *ACL*, 2004, pp. 128-135.

[59] B. Ribeiro-Neto and R. Baeza-Yates, *Modern Information Retrieval*, 2nd ed. Addison-Wesley-Longman Publishing co., 1999, http://people.ischool.berkeley.edu/~hearst/irbook/1/node2.html.

Appendix A

Project Plan Gantt chart

Appendix B

C-Value algorithm

Linguistic part of C-value:

- Tag the corpus using a tagger
- Extract strings using linguistic filter
- Remove tags from strings
- Remove strings below frequency threshold
- Filter rest of strings through stop-list

Statistical part of C-value:

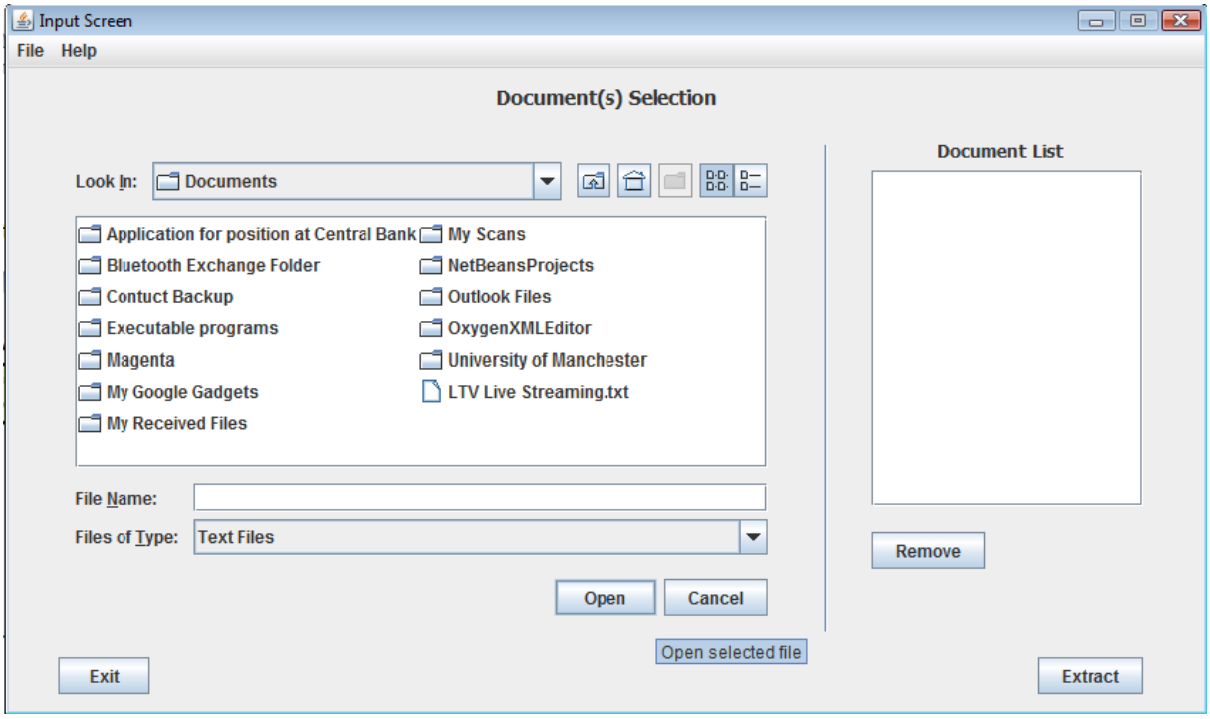
- For all strings *(a)* of maximum length
  Calculate *C-value(a)* = $log_2|a| \cdot f(a)$;
  If *C-value(a)* ≥ *Threshold*
  Add *(a)* to output list;
  For all substrings *(b)*
  Revise t*(b)*;
  Revise c*(b)*;
- For all smaller strings *(a)* in descending order
  If *(a)* appears for the first time
  C-value(a) = $log_2|a| \cdot f(a)$;
  Else
  C-value(a) = $log_2|a| \cdot \left(f(a) - \frac{1}{C(a)} - t(a)\right)$;
  If *C-value(a)* ≥ *Threshold*
  Add *(a)* to output list;
  For all substrings *(b)*
  Revise t*(b)*;
  Revise c*(b)*;

C-value algorithm as described in [1]

Appendix C

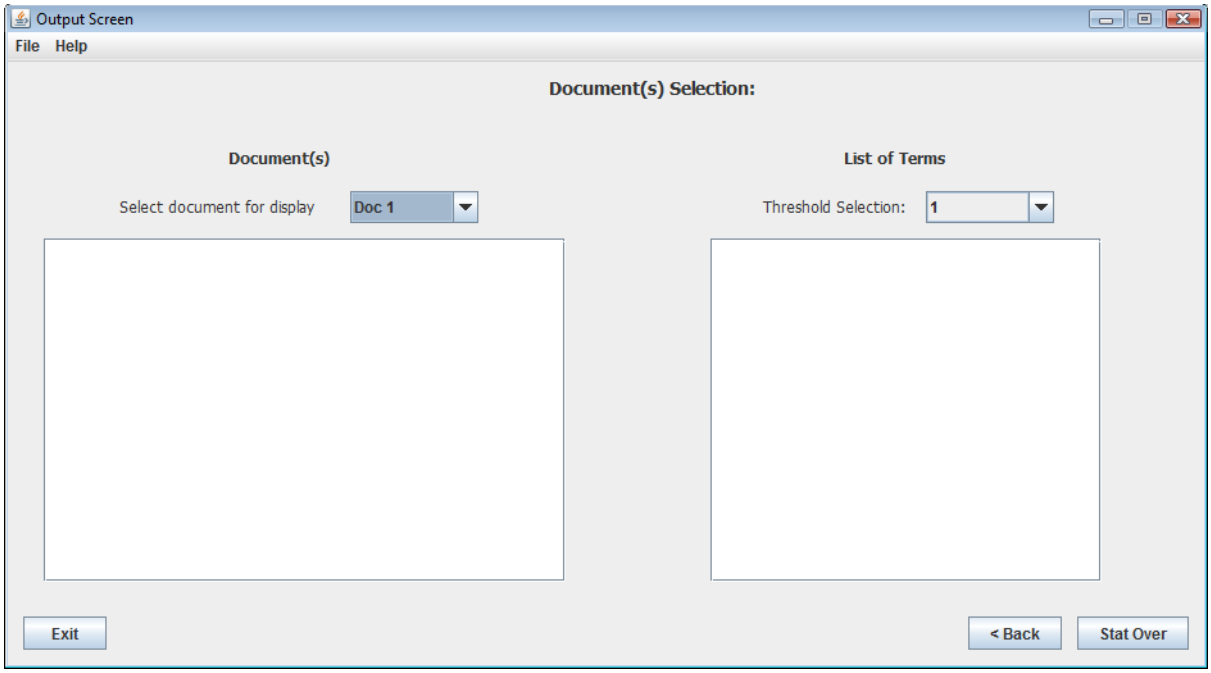
Screenshots of the 1st prototype

Screenshot1: Welcome Screen



Screenshot2: Input Screen

APPENTIX C



Screenshot3: Output Screen