Dana Dannélls
cl2ddoyt@cling.gu.se

Acronym Recognition

Recognizing acronyms in Swedish texts

October 16, 2006

A thesis submitted in partial fulfilment of the requirements for Master degree
in Computational Linguistics at the Department of Linguistics
Göteborg University, Sweden (corresponds to 30 ECTS credits)

Supervisors:
Lars Borin, Department of Swedish, Göteborg University
Dimitrios Kokkinakis, Department of Swedish, Göteborg University

Abstract This thesis addresses the task of recognizing acronyms and their definitions in Swedish text, primarily in the biomedical domain. The goal of this work is to build an electronic dictionary which will automatically grow with time.

Approaches from previous work on the automatic acronym recognition task are investigated and compared. A new approach is presented. Using this approach an automatic system, capable of extracting acronym-definition pairs was developed. The system uses general heuristics to identify and extract acronym-definition candidates, followed by a machine-learning algorithm which can be trained to classify the extracted acronym-definition candidates. As a result of successful classification, acronym-definition pairs are added to the database.

The identification process is based on part-of-speech tags, punctuation, and other textual markers. It does not rely on letter matching, nor on acronym-definition patterns. In contrast to pattern matching approaches, features characterising the acronym-definition pairs are used by the machine-learning algorithm. The features used are numeric and represent orthographic variations and morphological clues.

The system is structured in a way that allows further training and testing of the machine-learning algorithm. It can be used to experiment with different datasets in order to improve performance.

The system is partially available online for further research. The information stored in the acronym database will subsequently be accessible by electronic search engines.

Acknowledgements I take this opportunity to record my gratitude to all individuals, directly or indirectly, who have contributed towards the completion of this thesis.

First, I would like to express my heartfelt gratitude to my supervisors Lars Borin and Dimitrios Kokkinakis for their support and guidance throughout this work.

I would like to thank all my friends and students at the Department of Linguistics. I would also like to thank Anders Berglund for his constructive review of this thesis. Special thanks to Torbjörn Lager for his constant encouragement.

I would like to thank Robert Andersson for his valuable technical support. Thanks to Geoffrey Shippey for helping with English. Thanks to Walter Daelemans, David Nadeau and Peter Turney for testing the data and reporting the results.

Finally, I would like to thank my parents, my husband, and all my family members for their continuous support and encouragement.

Contents

List of Figures

List of Tables

❖ 1

Introduction

This chapter gives an introduction to the subject of the thesis and argues for its importance. The chapter also outlines the structure of the thesis.

1.1 Motivation

The growth of new acronyms in biomedical literature (Cheng 1994) increases the need for domain-specific dictionaries covering the new terms. This is important, since it is necessary to resolve acronyms particularly when there is no explicit definition in the text. A missing definition can lead to confusion regarding sentence meaning which in turn can hinder readers from understanding important concepts.

Building an acronym dictionary is a time-consuming and tedious process that requires many person-hours of effort by highly-skilled people (lexicographers) with extensive experience in the specific domain. Construction of an automatic domain-specific system that is able to extract new acronyms and present them with their related meaning is therefore crucial. Furthermore such system would help researchers and lexicographers with their time-consuming work and increase the availability of acronym dictionaries in both printed and electronic form. It will also facilitate natural language processing (NLP) such as co-reference resolution for named entity recognition.

The task of identifying and extracting acronyms has been studied over the last few decades, using different technologies from NLP with promising results. Today there are many electronic acronym dictionaries as components in NLP systems, freely available on the World Wide Web as searchable databases. These systems have the advantage of automatically identifying and extracting acronyms in free texts. They use different methods to scan text for acronym candidates, and then apply an algorithm to match them with their definitions in the surrounding text (some of these methods are presented in Chapter 3). However a consistent method covering the large variety

of different acronym formations is not available.

Automated methods for recognizing and defining acronyms play a major role in many NLP applications. The acronym recognition task is a big problem in the English language, and an even a bigger one in other languages such as Swedish (the problems exhibited in the Swedish language are exemplified in Section 4.2).

This thesis concentrates on the development of a robust, portable and trainable system which captures the acronym formation variations exhibited in Swedish medical texts. However the proposed method should have broad applicability to other languages and other domains as well.

1.2 Research questions

The questions I hope to answer in this thesis are: which of the existing methods is the most suitable for the task of recognizing and extracting Swedish acronyms? how can this method (or these methods) be used successfully to make non-trivial predictions on new data and avoid inconsistency?

1.3 Research goals

The primary goal of this research is to build a system capable of automatically collecting acronym-definition pairs from Swedish medical texts, for the purpose of constructing a Swedish acronym-definition dictionary. The system should be robust, fast, and effective.

1.4 Structure of thesis

The rest of this thesis is structured as follows:

Chapter 2: Terminology presents some basic linguistic concepts which are important for understanding the objectives of this research.

Chapter 3: Acronym systems and methodologies continues with a review of previous work. This chapter summarizes the methods used by different authors dealing with the acronym recognition task.

Chapter 4: Acronyms in Swedish explores the different challenges presented by Swedish and English acronyms. This chapter explains how the Swedish acronym-definition task

is different from the English one. In addition, previous systems are tested and evaluated on Swedish data.

Chapter 5: Automatic acronym recognition sets out the objective for this research, and describes how it should be achieved. This chapter introduces the tools included in the system.

Chapter 6: System overview describes the design of the acronym recognition system and the methods used to achieve the goals of this research.

Chapter 7: Evaluation and results evaluates the performance of the system and presents the results obtained with the machine-learning algorithm.

Chapter 8: Discussion discusses the achievements of this work and compares them with results using previous systems.

Chapter 9: Conclusions summarises the capabilities and the limitations of the system, and presents the achievements of this work.

Chapter 10: Future work directions for future work are suggested. In addition, some reflections on the possibilities for continuing development.

❖ 2

Terminology

This chapter presents some basic linguistic concepts and relevant terminology.

2.1 Acronyms

Acronyms are a subset of abbreviations[1] and are generally formed with capital letters. A distinguishing characteristic of acronyms is their lack of symbols such as apostrophe (') and fullstops (.). According to the *Oxford English Dictionary* (Simpson and C.Weiner 1989) the term *acronym* is defined as follows: "Acronyms are a type of abbreviation made up of the initial letters or syllables of other words".

Acronyms normally appear with common constructions such as 'NASA' from "National Aeronautics and Space Administration". However there exist acronyms which differ from this structure and do not follow the above mentioned definition. Examples are shown later in this section.

An acronym is a result of taking a phrase and shortening it into a new form. As with abbreviations, every acronym has a short form (SF) and a long form (LF), also called an expansion or definition, for example 'JAMA' (SF) is an acronym for "Journal of the American Medical Association" (LF).

Acronyms have wide coverage, especially in biomedical texts and may appear in different surface forms, i.e. letters in the acronym do not always match the initials of words in the definition. There are cases where letters in the acronym do not involve any letters similar to letters in the definition.

Acronyms include orthographical variations, such as different capitalizations or spellings,

[1] An abbreviation is a shortened form of a word, phrase, or symbol such as "wouldn't" from "would not".

for example an acronym variation for “retinoic acid receptor alpha” may be ’RAR alpha’, ’RAR-alpha’, ’RARA’, or ’RARa’.

An acronym can furthermore be of any length and does not necessarily consist of upper-case letters, e.g. ’apo’ from “apolipoproteinerna”. Acronyms can also be recursive, for example ’GNU’ from “GNU’s Not Unix” where the definition includes the acronym itself.

Two types of acronyms appear in biomedical articles (Yu et al. 2001). The first type are acronyms with an accepted definition and defined in standard vocabulary resources, such as the Medical Subject Headings (MeSH),[2] and the Unified Medical Language System (UMLS).[3]

The second type are acronyms which are defined within a particular text. These acronyms usually have definitions which vary depending on the authors who define them. Such acronyms can be found in comprehensive databases such as AcroMed, SaRAD, AGRH (these are discussed in Chapter 3) which are currently available online. For example, the following definitions were found for the acronym ’CHI’:
1. “Cambridge Healthtech Institute”, according to the Human Genome Acronym List.[4]
2. “Communication Hardware Interface”, according to the Internet Acronym Server.[5]
3. “Computer-Human Interface” (among the seven different definitions found), according to the Acronym Search database.[6]

When an acronym and its definition are introduced in a text for the first time, one of them usually appears within parentheses, according to one of the following patterns:
(1) definition (acronym);
(2) acronym (definition).

However there are cases where neither the acronym nor the definition appears within parentheses. In some cases, other symbols such as ’=’ and ’-’ are used to relate an acronym to its definition (shown in Table 5.1) but there are also examples of acronym-definition pairs which do not correspond to any of these patterns.

[2] http://www.nlm.nih.gov/mesh/
[3] Unified Medical Language System, a database containing biomedical information and a tools repository developed at the National Library of Medicine to help health professionals as well as medical informatics researches.
[4] http://www.ornl.gov/sci/techresources/Human_Genome/acronym.shtml
[5] http://silmaril.ie/cgibin/uncgi/acronyms
[6] http://www.acronymsearch.com/index.php

Acronyms are ambiguous. There can exist multiple definitions for the same acronym, for example 'CHI'. There can also exists multiple acronyms with the same definition. These present difficulties both for humans and NLP systems. The problem requires solutions that include semantic knowledge, and information about the context in order to identify the correct sense of the acronym when its definition is absent in the text.

Before applying disambiguation solutions it is necessary to have a complete representation of acronyms and definitions. A task which is addressed in this thesis.

2.2 Information Extraction

The current growth of biomedical knowledge has increased interest in NLP applications such as Information Extraction (IE), to cope with this increasing volume of biomedical articles.

IE is concerned with extracting relevant data from a collection of documents. IE systems rely on a set of extraction patterns that they use in order to retrieve from each document the relevant information. A key component of any IE system is its set of extraction patterns that is used to extract from each document the information relevant to a particular extraction task. Here we deal with an extraction task, where the important facts to extract from free biomedical documents are acronyms with their related meaning.

IE depends on term identification as one of the most crucial steps for accessing information stored in the literature. The goal of term identification is to recognize the term and capture its underlying meaning.

There are three main steps in the successful identification of terms from biomedical literature: term recognition, term classification and term mapping (see Figure 2.1).
Term recognition differentiates between terms and non-terms and finds the term boundaries. **Term classification** assigns terms to classes, such as genes and proteins. **Term mapping** links terms to well-defined concepts of relevant data sources, such as databases.

Here we address the first and the second steps in the identification process, which are to recognize and classify acronym-definition pairs.

Acronyms pose an interesting challenge to information extraction systems for several reasons. Existing resources do not provide a complete coverage for all the acronyms in the biomedical domain, mentioned in Section 1.1. Another reason is related to

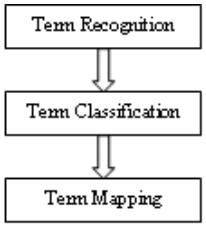

Fig. 2.1. Term identification steps.

the acronym disambiguation problem. It is necessary to have a complete coverage of acronym definitions and their variations in order to map acronyms to a correct definition when there is no explicit definition in the text.

2.3 Term recognition techniques

2.3.1 Automatic Term Recognition

The main goal of Automatic Term Recognition (ATR) is to identify terms in texts systematically and associate them with their concepts in an ontology.

Automatic recognition systems often rely on the identification of concepts, linguistically represented by domain specific terms. Such systems are important for maintaining consistency and thereby avoid terminological confusion especially because new terms are constantly emerging and undergoing changes in meaning.

One important aspect in the recognition process is orthographic and lexical variations, for example 'NF-kappaB', 'NF kappa B', 'NF(kappa)B', 'kappaB', 'NFKB factor', 'NF-KB' and 'NF kB' are all acronyms for "nuclear factor kappa B" (Spasic et al. 2005).

Term recognition systems are enhanced with the help of pattern, linguistic and statistics based approaches. Typically they are evaluated in terms of precision and recall. ATR methods normally rely on text pre-processing, such as normalization, and annotation such as part-of-speech (POS) tagging, and are mostly concentrated on Noun Phrases (NPs) since these cover the vast majority of terms.

2.3.2 Named Entity Recognition

Named Entity Recognition (NER) refers to the task of recognizing names and terms of biomedical entities, such as genes, proteins and diseases. These terms frequently consist of acronyms. NER is concerned with finding patterns and regularities in text

documents. The primary goal of NER is to relate each name entity of importance in a text document to an individual in the real world. In this regards, NER is different from ATR.

The approaches that have been applied to this task include dictionary-based, rule-based, machine-learning, and hybrid approaches. The main reason for using machine-learning techniques is that they are generally robust, can be retrained easily on new data, and successfully classify unknown examples.

These are comparable with the approaches dealing with the acronym task which share the same goal of an accurate identification of terms and their definitions in a body of text. However the approaches to biomedical NER have focused almost exclusively on gene and protein names, and hence may not be applicable directly to the acronym task.

2.4 Memory-Based Learning

Memory-Based Learning (MBL) is a machine learning paradigm which is used to detect patterns and regularities in a data set. The learning process is done by reading training instances into memory, and classifying test instances by extrapolating a class from the most similar instance(s) in memory. Memory-based learning (also known as instance-based learning) algorithms are sometimes referred to as “lazy” learning algorithms since they delay processing until new instances must be classified (Mitchell 1997). The advantage of these learning algorithms is that they estimate the target function locally and differently for each new instance to be classified.

Another advantage of memory-based learning is that it can handle successfully domains that present exceptions and irregularities (usually considered noise by other machine learning techniques).

2.5 Evaluation measures

Evaluating the performance of acronym recognition algorithms requires a suitable gold standard, in which the acronyms and their definitions have been classified manually by human readers.

The gold standard allows evaluation of different algorithms according to three metrics: Recall, Precision and F-measure (also called F-factor). These are considered the standard evaluation metrics for comparing alternative techniques in the field of NLP.

Recall is a measure of how much relevant information the system has extracted from the text and is a measure of completeness. In this project, recall represents the proportion of "correct" acronym-definition pairs extracted by the system to the total number of "correct" acronym-definition pairs in the dataset.

$$Recall \ = \ \frac{number \ of \ correct \ pairs \ returned}{total \ number \ of correct \ pairs \ in \ file}$$

Precision, also known as accuracy, is a measure of how much information returned by the system is actually correct. In this project, precision represents the proportion of "correct" acronym pairs to the total number of extracted pairs.

$$Precision \ = \ \frac{number \ of \ correct \ pairs \ returned}{total \ number \ of \ pairs \ returned}$$

Recall and precision are inversely related, F-measure is a combined value of recall and precision. The combined measure uses a parameter β to balance recall and precision:

$$F \ = \ \frac{(\beta^2 + 1) \ RP}{\beta^2 \ R \ + \ P}$$

F-measure is a straightforward way of combining the measures of recall and precision. When β is one, precision, and recall are given equal weight, and when β is greater than one, precision is favoured. When β is less than one, recall is favoured.

❖ 3

Acronym systems and methodologies

The task of extracting acronym-definition pairs from biomedical literature automatically has been studied for the past few decades, using technologies derived from NLP. Virtually all this research has been concerned with texts written in English. Existing approaches fall into two broad categories: (1) *Pattern-based methods*; (2) *Statistical and machine-learning methods*. Several hybrid methods have been published using both kinds of methods in different ways.

This chapter describes the different methodologies in each category and presents some of the existing extraction systems that were developed to deal with the acronym recognition task. Each method is followed by an evaluation that summarizes capabilities and drawbacks, as reported by its the author(s).

The performance of the systems and the corpora which were used in each experiment are summarized at the end of each section. It is important to note that the performance of the different methods are not directly comparable, since the authors use different test data for evaluation.

3.1 Pattern-based methods

Most traditional methods for finding acronyms with their definitions are based on pattern matching. A pattern is a configuration that characterizes a class and can be driven by a static or/and a dynamic model.

The majority of methods (Yeates 1999; Schwartz and Hearst 2003; Larkey et al. 2000) utilize static pattern-based matching. These methods are based on manually developed matching rules, typically very specific, for identifying certain acronym classes.

Several methods tend to combine general pattern-based approaches with Dynamic

Programming (DP) (Taghva and Gilbreth 1999), or with additional natural language processing (Pustejovsky et al. 2001) for supporting the recognition process.

Pattern-based methods differ from each other in the type of information encoded in their rules (patterns), which are crucial for the system performance.

Some approaches use dynamic models (Yeates et al. 2000), thereby avoiding manual development of pattern rules. In such models, rules are generated automatically from the examples found in the text.

3.1.1 Static models

Static models use a set of pre-defined criteria and are based on common patterns that appear in different texts. Such models may be specific or generalized. A general model has the advantage of capturing many of the structure variations that occur in a certain text while a specific model tends to capture specifies structures.

AFP - Acronym Finding Program

The Acronym Finding Program (AFP) presented by Taghva and Gilbreth (1999) was developed to identify acronyms and their definitions within an Optical Character Recognition (OCR) environment. Their program utilizes the Longest Common Subsequence (LCS) algorithm (Cormen et al. 1990) to find all possible alignments between the acronym candidate and the definition string, and is based on an inexact pattern matching.

Patterns: All upper-case words of three to ten characters are accepted as acronym candidates. Every letter in the acronym string must match with the first letter in each word of the definition string.

The algorithm

The AFP consists of four phases, the first two phases are used for identifying acronym candidates, and for determining a window in which a definition can possibly be found. The window size is limited to 2* $|A|$.[7]

The third phase is used to build a LCS-matrix by classifying words into: (1) stop words[8]; (2) hyphenated words; (3) normal words (words that don't fall into any of the above categories); (4) the acronyms themselves (since an acronym can sometimes

[7] $|A|$ is the number of characters in the acronym.

[8] The term "stop words" refers to prepositions, determiners, and conjunctions such as "the", "of", "and", which in this thesis are defined as "noise words".

be a part of the definition).

In the fourth phase, the algorithm calculates a heuristic score for each competing definition by parsing the matrices and generates a vector for each acronym and definition candidate. The last part of the algorithm compares and selects the pairs with the highest score.

Evaluation
The method considers only upper-case letter strings as potential acronym candidates, two letter acronyms are not handled by the system. If there is no first letter match for each word in the definition e.g. *Teledyne Wahchang Albany (TWCA), dioctyphthalate (DOP)*, or if there are too many words in the definition string without a corresponding letter in the acronym string, e.g. *grain boundary sliding controlled by lattice diffusion (GBL)*, the acronym pair falls below the LCS threshold, and is not considered a valid candidate. The algorithm parses words with embedded punctuation as single words and thereby fails to recognize acronym pairs such as *U.S. Geological Survey (USGS)*.

TLA - Three Letter Acronyms

Three Letter Acronyms (TLA) presented by Stuart Yeates (1999) was developed to extract acronyms in technical and governmental documents. The algorithm uses general heuristics to match characters of the acronym candidate with letters in the definition string. As with AFP, acronym candidates and their definitions are selected from a stream of raw text.

Patterns: Potential acronyms must be entirely in upper-case and shorter than their definition, since short acronyms tend to have longer words in their definitions. Acronyms must contain the initials of most of the words in the definition string.

The algorithm
All non-alphabetic characters are converted to spaces and any multiple spaces replaced with a single space. Acronym candidates are determined by matching the initial letter of each word in the context with a letter in the potential acronym.

A maximum of six letters in the acronym are compared against each word in the definition string. If the first letter does not match, the word is skipped. Otherwise, the next letter of the same word is tested against the next letter of the acronym, the algorithm continues to move along the word as long as there is a letter match. The resulting acronyms are sorted, and duplicates removed.

Evaluation
TLA does not recognize abbreviated expressions containing more than one upper-case letter from the acronym e.g. *DataBase Management System (DBMS)*. Three letters in the acronym string are tested against each word in the definition string. All of the predefined patterns must be satisfied before an acronym candidate is accepted, a limitation which excludes acronyms such as *apolipoprotein B (apoB)*.

Acrophile

Acrophile (Larkey et al. 2000) is an automated system that builds and serves a searchable database of acronyms and abbreviations gathered from a large number of web pages. The system uses information retrieval techniques and ad-hoc heuristics to extract additional acronym-definition pairs, and collects them in a database.

Four different algorithms were implemented and compared: contextual, canonical, canonical/contextual, and simple canonical. The algorithms were tested and evaluated separately, they differ in the patterns that are used to find potential acronyms. The algorithms and their results are summarized in Table 3.1. The authors concluded that the canonical/contextual algorithm is the most accurate.

Patterns: The contextual algorithm accepts acronyms that are all upper-case, they may include periods, digits or have a sequence of lower-case characters either at the end of the string or internally following at least 2 upper-case characters such as 'AChemS'.

The canonical/contextual and canonical algorithms allow acronym strings which contain a mixture of lower and upper-case letters e.g. *Department of Defence (DoD)*, periods, one non-final digit, slashes, and hyphens e.g. *AFL-CIO (American Federation of Labor - Congress of Industrial Organizations)*. A definition candidate is accepted if it occurs within parentheses or/and comes directly after/before the acronym.

There are only a few expressions which are allowed between an acronym and a definition such as "A stand for D", "D known as A". Noise words which are normally skipped in the acronym are allowed in the definition. Hyphenated definition terms such as "Real-Time" are treated as one or two words without requiring the character 'T' to appear in the acronym.

The simple canonical algorithm only accepts acronyms that begin with an upper-case letter, followed by zero to 8 upper or lowercase letters, slashes, or dashes and ending in an uppercase letter.

The algorithm

The algorithms identify potential acronyms by scanning the text according to the patterns defined for each algorithm. The search space for a definition consists of twenty words (adjacent to the acronym).

Each algorithm tries to match multiple characters in the definition string starting from the last character of the acronym toward the beginning. If the first character of an acronym is a digit the acronym receives special handling such as digit replacement, e.g. '3' is replaced with "three", and the algorithm tries to match the spelled out number with a word in the definition. A potential definition string is scanned if there is a pattern that allows matching, or until the definition for the potential acronym is found.

Evaluation

For the canonical/contextual algorithm acronyms are not allowed to end with lowercase characters except for 's', and only 1 digit is allowed in the acronym string, thus it fails to recognize acronyms such as *phorbol ester 12-myristate-13-acetate (PMA)*. Lower-case words are allowed in the definition only if the acronym pair matches one of the pre-defined patterns. The pre-defined patterns do not allow acronym-definitions with the following appearance in the context: *trophectoderm* is the definition of *TE*.

The contextual canonical, and simple canonical algorithms accept fewer patterns for acronym-definition candidates.

Algorithm	Precision	Recall
contextual	89 %	63 %
canonical/contextual	87 %	88 %
canonical	96 %	60 %
simple canonical	94 %	59 %

Table 3.1. Acrophile algorithm performance.

AcroMed

The AcroMed system (Pustejovsky et al. 2001) is a part of a set of information extraction tools designed for processing and extracting acronym-meaning pairs from abstracts in the MEDLINE database. The Acromed utilizes syntactic information and regular expression patterns to match acronyms and find their meaning in the context.

Patterns: The regular expressions only allow acronym-definition pair candidates according to one pattern: "Definition(Acronym)". The first word of the definition string

must use the first letter of the acronym string.

The algorithm
A potential acronym is converted into a regular expression which is used to capture its definition within a noun phrase by searching the left context where the acronym was found. When a string (definition candidate) is found it is rated with a formula that compares how good the acronym is in relation to a comparison or threshold measure.

The algorithm tries to match the acronym with a prefix or infix of the words that compose the definition string candidate. If there is a match the algorithm scores the acronym-definition pair according to the number of matches (similar to Taghva and Gilberth's approach). If the score is below the threshold (1.5) the pair is accepted, and added to the acronym database. If no definition was found the meaning of the acronym is looked for in the acronym database.

Evaluation
An acronym must appears within parentheses. The algorithm only seeks for definitions by looking to the left side of the acronym string, thus an acronym must always follow its definition.

The AcroMed fails to identify acronym-definition pairs, such as *helper T (TH)*. Some errors in the acronym-definition pairs are related to the problem of assigning a window or boundary for the search of the definition of the acronym, e.g. the algorithm found *p16, products* in the following context: " . . . which encodes two gene products (p16(INK4a) and p19(ARF))".

A simple algorithm

Schwartz and Hearst (2003) present a simple algorithm for extracting abbreviations and acronyms (short forms) together with their definitions (long forms) from biomedical documents. The algorithm identifies short forms and matches their letters with those in the long forms.

Patterns: An acronym candidate contains between 2 to 10 characters, it consists of at most two words, at least one character is a letter and its first character is alphanumeric.

A definition string must appear immediately before or after the corresponding acronym, i.e. in the same sentence and should have no more than ($|A|$+5) or ($|A|$*2) words.

The algorithm

When an acronym candidate is found the algorithm searches its shortest definition string by scanning the left, and right context. It tries to match each character in the acronym with a character in the definition by moving to the left, starting from the end of both strings.

The algorithm succeeds in finding a correct acronym-definition pair if the first character of the acronym string matches the first character of the word in the definition string.

Evaluation

Only two patterns are considered by the algorithm, viz "Definition(Acronym)" and "Acronym(Definition)". A definition must be adjacent to the acronym. The algorithm fails to identify acronym-definition pairs if there is no exact character match in the definition string, e.g. *CNS1 (cyclophilin seven suppressor)*, and/or if characters do not appear in the same order as in the definition string, e.g. *ATN (anterior thalamus)*. Definitions which are shorter than their corresponding acronym, or which include the corresponding acronym within one of their single words are not considered as pair candidates.

3.1.2 Dynamic models

Dynamic models use a set of criteria that are generated automatically according to patterns found in a text. The generated patterns in a dynamic model may be refined on new texts. Such models have the advantage of being applicable to different texts without modifications.

Compression

Compression-based methods can be defined as a pattern-based method that uses a dynamic model to find patterns and regularities in a text.

Yeates et al. (2000) present a compression-based algorithm for coding potential acronyms with their definitions.

Heuristics: An acronym candidate contains least two upper-case letters. An acronym-definition pair candidate is considered valid if the ratio between the number of bits required to code it (according to the text model) is less than a certain proportion of the number of bits required to code it in the context (according to the general compressor).

The algorithm

The algorithm filters the text by identifying acronym candidates and determines two search windows for the definition. Each window contains 16 words, one for the words preceding the acronym candidate, and one for the words that follow the acronym candidate. The windows are used to compress each acronym candidate by coding it with respect to the initial letters in the window.

Four components are used to compress the acronym with respect to its context: (1) whether the acronym proceeds (+) or follows (-) the definition; (2) the number of words between the acronym, and the definition; (3) the sequence of words in the text; (4) the number of characters taken from each word. As a result a string that contains the matched words and letters is produced. For example: for the acronym 'BC' that appear in the context "Both the Bandwidth Constraction (BC) algorithm", the following string is produced: "- 2 $\langle 1 \rangle$ $\langle 1, 1 \rangle$".

Each acronym string is then compared and the string that compresses best is selected. A text model is then used to compress the acronym-definition pairs, and to determine whether it is valid by calculating the number of bits required to represent the characters in the acronym.

Evaluation

Hyphens and slashes are included as word boundaries. The coding scheme does not permit acronyms which include digits e.g. *Binary 8 Zero Substitution (B8ZS)* and which appear in plural forms, e.g. *Collaborative Virtual Environments (CVEs)*. The algorithm requires a corpora of hand-marked training data.

System/Method	Precision	Recall	F-score	Corpus
AFP	98 %	93 %	95.4 %	17 technical documents
TLA	68 %	91 %	77.8 %	10 technical reports
Acrophile	87 %	88 %	87.5 %	353 web documents
AcroMed	98 %	72 %	83 %	155 MEDLINE abstracts
Simple algorithm	96 %	82 %	88.4 %	168 MEDLINE abstracts
Compression	90 %	80 %	84.7 %	1080 technical reports

Table 3.2. Pattern-based algorithm performance.

3.2 Statistical and machine-learning methods

A variety of statistical and machine-learning techniques are used for different identification problems and have proven their efficiency in several NLP tasks.

Several authors applied statistical-based techniques (Chang et al. 2002; Adar 2002) to the acronym recognition problem. Their methods use statistical information based on the frequency of acronyms in corpora. These methods are also combined with machine-learning approaches.

Learning algorithms use examples, attributes and values to recognize and classify new terms. They have the capability of improving with experience (trainable) and making non-trivial predictions on new data. Machine-learning techniques are particularly useful for medical researchers since biomedical records often involve many features and many examples.

Authors who applied machine-learning algorithms to the acronym recognition problem (Nadeau and Turney 2005) define the task in terms of a learning problem[9]. The concept we want to learn [T] is a pair (A, D) consisting of an acronym A (a single token), and a definition D (a sequence of one or more tokens). Given a sequence S of n tokens, S = ($s_1 \ldots s_n$), from which we wish to extract a pair (A,D). Performance [P] improves as the percentage of correctly classified pairs. The training experience [E] is a sequence of tokens with given classified pairs.

Stanford: Online dictionary of abbreviations

Chang et al. (2002) present a robust method for identifying abbreviations using supervised machine-learning. The method utilizes linear regression on a pre-selected set of features (Appendix B.1) that describe different patterns seen commonly within abbreviations.

Heuristics: The algorithm only considers acronym-definition candidates which appear between parentheses according to one pattern: “Definition(Acronym)”. An abbreviation must contain a letter. The search space for a definition string is 2*$|A|$ words, and is between the found abbreviation up to a comma or a semicolon.

[9] “A well-posed learning problem is defined as follows: A computer program is said to learn from experience E with respect to some class of tasks T, and performance measure P, if its performance at tasks in T, as measured by P, improves with experience P” (Mitchell 1997)

The algorithm
When abbreviation candidates are found, the algorithm aligns them to the preceding text using the LCS algorithm. The acronym-definition candidates are converted into feature vectors, which are then scored by a statistical machine-learning algorithm. To determine whether a acronym-definition candidate is true, a binary logistic regression classifier is trained on a hand-annotated set of MEDLINE abstracts.

Evaluation
Abbreviation candidates longer than two words, and that exactly matching the words in the preceding text are rejected. The trained classifier is used to handle continuous data.

SaRAD - A Simple and Robust Abbreviation Dictionary

The Simple and Robust Abbreviation Dictionary (SaRAD) presented by Eytan Adar (2002) is similar to the method described by Chang et al. (2002).

Heuristics: An acronym is a word or hyphenated compound words where there is at least one capital character. A definition candidate must precede the acronym and its length should not exceed *n*, where *n* is $2 * \mid A \mid +$ the number of words before the parenthesis.

The algorithm
When an acronym is found, the window for the definition is determined, and its characters are normalized to lower-case characters.

The algorithm searches forward through this text window, and attempts to match each character with the characters in the acronym. This generate a set of definitions candidates. To find the correct definition each candidate is scored according to its match, i.e. first letter match, position in the text window. If the score of a candidate is higher than the threshold (0) it is considered a valid definition candidate.

The next step is to convert the words in the definition candidates by using tri-gram clustering, for example the string 'ABCDE' is converted to {ABC,BCD,CDE}. A similarity method is then used to compare the definition candidates to each other, and to find the most common definition for each cluster, this way allows the definition "cholecystokinin" to match with its valid acronyms: 'CCK', 'CCK-33' and 'CCK-39'.

Evaluation
Only candidates which appear between parentheses according to the pattern: "Defini-

tion(Acronym)" are considered. Complicated acronym-definition patterns tend to get a low score and are rejected, e.g. *Gen-5 Related N-acetyltransferase(GNAT)*.

A supervised learning approach

Nadeau and Turney (2005) present a supervised learning approach to the acronym identification task.

Heuristics: An acronym candidate is a token that consists of 1- *n* alphanumeric characters and may include punctuations. The first word of the definition must use the first letter of the acronym. The definition must not contain a bracket, colon, semi-colon, question mark or exclamation mark. Maximum length is min$(\mid A \mid +5, \mid A \mid *2)$.

The algorithm

A set of heuristics is used to find acronym candidates, and to limit the search space for the definition. The acronym candidates are converted into vectors, each vector consists of seventeen features (Appendix B.2) describing each member of the acronym pairs. To determine whether the acronym-definition pair is a true candidate the algorithm tests against an annotated corpus. The acronym-definition pair is labelled as positive if there is an exact match of the acronym and the definition string in the corpus.

Evaluation

The weak heuristics allow the identification of a large number of acronym-definition candidates which results in high recall. However there are a number of acronym-definition pairs that the algorithm fails to identify, such as *cyclophilin seven suppressor (CNS1), cAMP-dependent protein kinase A (PKA), Thyroid transcription factor 1 (TTF-1)*.

As the authors point out, the number of features may be reduced since not all of these contribute to the learning process.

System/Method	Precision	Recall	F-score	Corpus
Stanford	80 %	83 %	81.5 %	168 MEDLINE abstracts
SaRAD	95 %	85 %	89.7 %	155 MEDLINE abstracts
Supervised learning	89 %	88 %	88.5 %	168 MEDLINE abstracts

Table 3.3. Statistical and machine-learning algorithm performance.

Hybrid methods

Hybrid text mining

Park and Byrd (2001) present a hybrid method for processing abbreviations in text documents. Their method employs pattern-based rules, syntactic structures (special symbols such as '(..)', '[..]') and linguistic cue words (includes words such as: "or", "short").

Heuristics: An abbreviation candidate is a string of alphabetic, numerical, and special characters. Its length is between 2 to 10 characters, and it contains at least one capital letter. The maximum length of a definition is min $(\mid A \mid +5, \mid A \mid *2)$. The first and the last word in the definition are not prepositions, be-verbs, modal verbs, conjunctions or pronouns and none of the words in the definition includes symbols like '\', '=', '!', '?'.

The algorithm
When an abbreviation candidate is found, the algorithm determines the context within which to look for the definition. The search space for the definition is its maximum length + 10 words to the right and to the left side of the abbreviation candidate.

When pair candidates are found, patterns which describe an abbreviation and a definition candidate are generated. Two characters are used to generate a string which describes the abbreviation candidate, these are: 'c' for alphabetic characters and 'n' for numeric characters. Five characters are used to generate a string that describes the words in the definition candidate, these are: 'w'(word), 's'(stopword), 'p'(prefix), 'h'(headword) and 'n'(number).
For example, the abbreviation pair ('X2B', "Hexadecimal to Binary")
is represented as ('cnc', 'phsw').

Abbreviation rules are then generated. These rules are used to describe how the abbreviation is formed from its definition. It consists of an abbreviation pattern, a definition pattern, and a formation rule.

A formation rule defines how each character in the abbreviation is formed from its definition. It consists of pairs, i.e. a word number (the location of the word within the definition) and a method. There are five formation methods: 'F' (first character), 'I'(interior character), 'L'(last character), 'E'(exact match for numeric characters) and 'R'(replacement match). For example, an abbreviation rule for the pair given in the last example is: $\langle' cnc',' phsw', (1, R), (3, R), (4, F)\rangle$.

The algorithm searches the rule-base for an abbreviation rule that successfully gen-

erates the pair pattern. If no rule match found in the rule-base, and the pair matches with one of the predefined syntactic structures it is tested against a set of five rules for determining the candidate definitions based on their length. If no match is found among the five rules, the candidate pair is discarded. If there is a rule match or the pair does not match with one of the predefined syntactic structures, a new rule is created and added to the rule-base.

A set of six heuristics determines which definition to choose if there are several candidates. These heuristics are: (1) syntactic cues; (2) rule priority; (3) distance; (4) capitalization; (5) number of words; (6) number of stop-words.

Evaluation
The algorithm uses a prefix list which includes prefixes, such as: “ethyl”, “hexyl” to recognize certain acronym-definition pairs, e.g. *di-2-ethylhexylphthalate (DEHP)*. Adaptation of the method to new technical domains involve the addition of domain-specific prefixes, to the prefix list. If there is no letter match between the abbreviation and the definition, the algorithm fails the identify them, e.g. *alanine aminotransferase (ALT)*.

Other situations where the algorithm fails are: letters occurring in different order in the definition, e.g. *Italian Space Agency (ASI)*, letter is missing in the definition, e.g. *medium-class Explorer(MIDEX)*, and numeric replacement, e.g. *Third Convection and Moisture Experiment(CAMEX-3)*.

Performance As reported by Park and Byrd, the algorithm achieved a precision of 97% at recall of 94%, yielding F-score of 95.9%. They used a collection of 177 engineering texts as the corpus in their experiment.

❖ 4

Acronyms in Swedish

Acronyms are universal phenomena and are present in the written form of most languages. Although most researchers have focused on the English language, it is of great interest to discover acronyms similarities and differences in languages other than English.

This thesis is concerned with the Swedish language, and this chapter describes problems and difficulties with acronym-definition pairs appearing in Swedish texts. These are compared with problems in English.

4.1 English

Within the biomedicine domain, the accessibility of terminological knowledge bases listing English acronyms with their associated senses is large and extensive. The existing resources (e.g. UMLS and MeSH) facilitate the development of computer systems which behave as if they "understand" the language of biomedicine and health. These knowledge sources assist developers with particular purposes such as solving the acronym disambiguation problem.

As will be shown in Section 4.3 below, the methods developed for automatic construction of abbreviation knowledge bases using MEDLINE abstracts presuppose parenthetical expressions. The use of parentheses is very common in MEDLINE abstracts. Some types of acronym appearing in the MEDLINE abstracts:

- An acronym consisting of initial letters of three different words in the right order, e.g. *minimum alveolar anesthetic concentration (MAC)*.
- An acronym containing a comma, digits and other symbols, e.g. '-'.
- A definition contains only lower-case letters, and contained within parentheses directly after the acronym, e.g. *HIV (human immunodeficiency virus)*.
- A definition consists of only one word, e.g. *carboxymethyllysine (CML)*.

- Most acronyms are multi-word units (compounds). A typical example in three letter acronyms, where the words in the definition includes two initial letters of two words and the remaining one appears in one of these two words in the right order followed by at least three letters, e.g. *procoagulant activity (PCA)*; *indirect immunofluorescence (IIF)*.

There are many problems concerning acronym formation structures in English. Some of the most common ones are:
(1) Matches of acronym letters with other than the initial letters of words in the definition.
(2) Skipped letters in the acronym, such as the letter 'P' in the following pair: *Asia Pacific Association for Machine Translation (AAMT)*.
(3) Noise words, such as 'for' in the previous example, where 'f' is not included in the acronym string. There are also counter examples where there actually exists a letter match for noise words in the acronym.
(4) Mixture of Latin and Arabic numbers, *Usher type III (USH3)* or numeric replacement, *HL7, Health Level Seven*.
(5) Mixture of lower and upper case letters, *acute bacterial meningitis (ABM)*.
(6) Irregular acronym structures which are hard to detect since they are less common, for example: *5-HT, serotonin*; *Pol I, RNA polymerase I*.

4.2 Swedish

Swedish has an alphabet which is very much like the English alphabet, with the notable exception that Swedish has three additional vowels: å, ä and ö.

Swedish has a significant number of acronyms. Definition strings are normally composed with more than two words. Compounding is a very common phenomenon in Swedish, also among acronyms, where up to five stems can be put together to build one word. The boundary between the stems is often easy to find since the 's' letter is often used to compose between two words. For example the noun: swe. "samhälls**s**liv", eng. "social life".

However the linking 's' can sometimes be left out, and there are many examples where word boundaries are not indicated, a fact which makes the recognition task much more difficult. For example consider the acronym: swe. **l***äkar***f***örbundet, MA*, eng. *Medical Association, MA*. In the English example, matching each letter in the acronym string with a letter in each word of the definition string is not difficult, compared to the Swedish example where there is neither a letter match nor word boundaries.

Problems with English acronyms mentioned in Section 4.1 can also occur in Swedish. Some examples are:
(1) An acronym does not always involve letter similarities with its definition, e.g. *kvalitetsjusterat levnadsår (QALY)*.
(2) Skipped letters in the acronym, such as the letter 'l' in the following pair: *propylene glycol alginate lösning (PGA)*.
(3) Noise words, *Institutionen för fysiologi och farmakologi (FYFA)*.
(4) Mixture of Latin and Arabic numbers, *Usher typ III (USH3)*.
(5) Mixture of lower and upper case letters, *magnetresonanstomografi (MR)*.
(6) Irregular structures, e.g. *neuralrörsdefekter, NTD*; *röda hund (MPR)*.

In addition to problems common to both languages, there exist further difficulties in Swedish, making the acronym identification task more complex, for example compounding and letter matching. As shown in the previous example (i.e. MA), solutions to these problems are not straightforward.

Ambiguity is a bigger problem with Swedish acronyms, mainly because many of the definition strings have bilingual appearance in the Swedish texts, i.e. Swedish texts include both the English, and the Swedish definition strings. For example, some of the definitions for the acronym 'CT' in English are: "Computer Tomography"; "Computed Tomographs"; "Computerised Tomographic". These definition strings can appear in Swedish texts in addition to the Swedish definition strings, e.g. *datortomografi*, which are translated from English.

The mixture of Swedish definitions with acronyms derived from English makes the automatic recognition task much more difficult, creating severe problems with letter matching. For example, the English definition for 'NSAID' is "nonsteroidal anti-inflammatory drugs", where there is an exact letter match for each word in the definition. In the Swedish definition "icke-steroida anti-inflammatoriska medel" for the same acronym, only two-letter match.

They are some definition strings in Swedish that are only superficially different from English and cause little difficulty, such as the definition string for the acronym 'USH3' mentioned above (example 4). Greater differences which increase the recognition problem, for example: eng. *Ventricular Fibrillation, VF*, swe. *kammarflimmer, VF*.

Another difference between English and Swedish is the fact that Swedish lexical resources are not available to the same extent as English ones. Annotated texts and electronic databases cannot be used directly by computer systems since standardized vocabularies are still under development.

Table 4.1 exemplifies some Swedish acronym-definition pairs and their common characteristics.

Acronym	Definition	Description
TWCA	**T**eledyne**W**ahchang **A**lbany	Definition letters are capitalized.
VOC	organiskt hjärtfel	Acronym letters that do not participate.
Ecv	Extra**c**ellulär**v**ätska	Aligned on a syllable boundary.
MR	**m**agnet**r**esonanstomografi	Mixture of lower and upper case letters.
NS	icke-**s**teroida	Aligned on a punctuation boundary.
mRNA	budbärar**RNA**	Aligned immediatly after another.

Table 4.1. Acronyms' common characteristics and variations.

4.3 Comparison of existing systems

One of the main difficulties with acronym recognition is their wide coverage, known as the acronym formation problem. As shown in Section 4.2 Swedish acronyms share many problems with English ones. Hence the existing methods presented in Section 3 should also perform well on Swedish data.

To test this assumption, and to get a more complete understanding of the differences between the challenges set by English and Swedish acronyms, four methods were tested on the un-annotated version of the Swedish dataset (described in Section 5.2).

There are two reasons for choosing these particular systems: 1) They are available online, so it is possible to test them on different datasets; 2) The methods presented in Chapter 3 cannot be compared directly, because of the lack of common evaluation metrics and test corpora. However, the four chosen systems were evaluated on the same corpus, using the same gold standard, and are therefore comparable.

The systems that were tested with their results are given in Table 4.2. These results are compared with the results reported when evaluated on the same set taken from the MEDLINE abstracts. The table is followed by an analysis commenting on the systems and their performance.

	MEDLINE abstracts			Swedish text		
System	Precision	Recall	F-score	Precision	Recall	F-score
Acrophile	87 %	88 %	87.5 %	97 %	20 %	33 %
Stanford	80 %	83 %	81.5 %	77 %	66 %	71 %
Simple algorithm	96 %	82 %	88.4 %	100 %	6 %	11.3 %
Supervised learning	89 %	88 %	88.4 %	96 %	91 %	93 %

Table 4.2. Comparison of existing systems.

Acrophile

The canonical/contextual algorithm[10] (Larkey et al. 2000) gave low recall (20%) when tested on the Swedish data. The low recall can be explained by the fact that all the acronyms found were upper-case words with an exact letter match with the first letter of the words in the definition string. An incorrect pair found by the algorithm is: *HHV, HSV HSV VZV*, Examples of the incorrect pairs found by the 'HHV' is not an acronym that appeared in the input text file.

Acrophile compared to the AFP: Larkey et al. compared their algorithm with the AFP algorithm. The algorithms were evaluated on 170 web pages. The results from the AFP showed 76% precision at 31% recall while the canonical/contextual algorithm gave 76% precision at 34% recall. Thus, the Acrophile performs better than the AFP since it was able to find more acronym candidates.

Acrophile compared to the AcroMed: The performance of AcroMed (Pustejovsky et al. 2001) was compared to Acrophile by Pustejovsky et al. showing that the AcroMed gave lower recall than the Acrophile. The result can be explained by the fact that the system was designed to deal with three-letter acronyms – understandable since three-letter acronyms are most frequent in the MEDLINE abstracts. It is therefore unlikely that the AcroMed will perform better than the Acrophile on the Swedish dataset.

Stanford

The performance obtained by Chang et al. (2002) algorithm are 77% precision at 66% recall. The algorithm found 737 acronym-definition pairs of which 569 were correctly identified.

Some examples of the incorrect pairs that were found are: (1) *CSIS, och strategiska studier* instead of *CSIS, Centret för internationella och strategiska studier*; (2) *gör, hörselspecifika parametrar*, the word 'gör' is a verb. The pairs that the algorithm missed are those with irregular structures such as *apo, apolipoproteinerna*.

[10] The canonical/contextual algorithm was tested since as it gives the best results, according to the authors, see Table 3.1

Simple algorithm

The algorithm (Schwartz and Hearst 2003) found only acronym-definition pairs which correspond to one structure, i.e. “Acronym(Definition)”. Giving a very high precision, 100% but very low recall, 6%. According to the authors the algorithm should have captured acronyms which appear within parentheses according to the pattern: “Definition(Acronym)”. Unfortunately no acronym-definition pairs corresponding to this pattern were included in the results.

Supervised learning

The algorithm presented by Nadeau and Turney (2005) gave the best performance on the Swedish dataset. However these results were obtained after training their model on the Swedish data. The partition used was 60% training, 40% testing. The results of testing the model without training were 67% recall at 96% precision. Most of the errors where due to wrong segmentation (part-of-speech tags).

Table 4.2 shows high precision can be achieved regardless of the original target language of the system. However it is crucial to take into account of distinguishing characteristics between languages to achieve good overall performance. Hence systems developed to deal with one language are not necessarily applicable to other languages.

These results support Zahariev’s assertion (Zahariev 2004). In his thesis he presents a universal explanatory theory of acronyms which is based on examples from fifteen languages, with six different writing systems. Zahariev argues that in many languages, acronym formation is governed by linguistic preferences, based on regularities at the character, phoneme, word and phrase levels. For example, there are certain regularities that are very productive only in languages where morphological compound words are common, such as German, Swedish, Dutch and Finnish. Thus if an automatic acronym recognition system performs well on a specific language, such as English, it will not necessarily give as good results with languages such as Swedish, German, Dutch and Finnish. On the other hand, if an automatic acronym recognition system performs well on the Swedish language, it might give as good results with related languages such as German, Dutch and Finnish.

To summarise the results of testing existing algorithms on the Swedish data, together with system comparisons made by other authors:

- Complex pattern matching algorithms should be avoided if the algorithm will allow many acronym pairs variations.
- High recall can be achieved by generalizing the pre-defined patterns. These should be general enough to capture many acronym-definition pairs.

- Syntactic information about the context increase precision. Both Nadeau and Turney (2005) and Park and Byrd (2001) use POS tags, Pustejovsky et al. (2001) use noun-phrases (see Acrophile and supervised learning in Table 4.2).
- A machine-learning approach is a suitable strategy to consider. As shown in Table 4.2 the supervised learning algorithm gave very good results when tested on the Swedish data.

❖ 5

Automatic acronym recognition

This chapter outlines the research objectives, and describes the techniques and tools used to accomplish them.

5.1 Research approach

The main research objectives are to automate acronym search in Swedish medical texts and expand an electronic acronym dictionary. The main purpose for automatic acronym extraction is to reduce the time consuming work of lexicographers.

To accomplish this objective an automatic system capable of extracting and storing acronym pairs has been developed. The system uses syntactic units and grammatical knowledge to identify and extract acronym-definition pairs. It allows the user to train a machine-learning classifier and use this classifier to classify new acronym-definition pairs. The system is described in details in Chapter 6.

The system utilizes: (1) a set of heuristics to identify and extract acronym-definition candidates. These are presented in Subsections 6.2.1 and 6.2.2; (2) a memory-based algorithm for training and subsequently predicting the class of acronym-definition pairs. The machine-learning algorithm is presented in Subsection 6.3.2.

5.2 The data

The data used in this work consists of a manually annotated set of 861 acronym-definition pairs. The set was extracted from Swedish medical texts, the MEDLEX Corpus (Kokkinakis 2006). The material has been annotated using the CADIXE XML Annotation Editor. [11] For the majority of the cases in the sample, there exists one acronym-definition pair per sentence, but there are cases where two or more pairs can be found.

[11] The CADIXE XML Annotation Editor is availabe at: http://caderige.imag.fr/

An example of a sentence taken from the MEDLEX annotated samples:

“Stegrade nivåer av tre tackykininer $\langle definition\ id = \text{”}1\text{”}\rangle$substans P$\langle/definition\rangle$ ($\langle acronym\ link = \text{”}1\text{”}\rangle$SP$\langle/acronym\rangle$) ,
$\langle definition\ id = \text{”}2\text{”}\rangle$neurokinin$\langle/definition\rangle$ ($\langle acronym\ link = \text{”}2\text{”}\rangle$NK$\langle/acronym\rangle$)A och NKB ses i bronkialsköljvätska . . . ”.

There are two acronyms annotated, ’SP’, and ’NK’ linked to their two definitions, ’substans P’, and ’neurokinin’.

The annotated acronym-definition pairs were extracted from this dataset and were merely used to evaluate the performance of the system (see Chapter 7).

Table 5.1 shows the distribution of the acronym-definition pairs in the Swedish dataset. Some examples include:
A = D: AVNRT = AV-nodal reentrytakykardi
A - D: ACE - Angiotensin Converting Enzyme
A (D): PAI-1 (plasminogen activator inhibitor 1)
D (A): reumatoid artrit (RA) .
A , D ,: tPA, tissue-type plasminogen activator ,
D , A ,: C-reaktivt protein , CRP ,

Pattern	#occurr.	%
D (A)	570	66.2 %
D , A ,	122	14.2 %
A (D)	49	5.7 %
D , A .	44	5.1 %
A , D ,	12	1.4 %
” D ”(A)	9	1 %
A = D	6	$<$1 %
A - D	4	$<$1 %
Rest	45	5.2 %

Table 5.1. Distribution of definitions (D) and acronyms (A) in the data set.

5.3 Programming language and software

The system is implementated in Java.[12] One purpose of the system is continual expansion of the database of acronym-definition pairs. This is stored in a mySQL relational

[12] The api is available at:
http://www.cling.gu.se/∼cl2ddoyt/acronymrecognition/api/

database. Users are able to interact with the system by submitting a query via CGI (Common Gateway Interface). The following subsections provide a description of the programs that the system integrates with.

5.3.1 Trigrams'n'Tags

Trigrams'n'Tags (TnT) is a statistical part-of-speech tagger developed by Thorsten Brants (2000). The tagger is based on Hidden Markov Model (HMM). It is adaptable to new languages, is trainable on different domains, and virtually on any tag set. It is necessary to train the tagger on similar texts of which the tagger will be applied to, in this case biomedical texts.

To train a suitable tagger, a dataset taken from the biomedicine domain was automatically tokenized and manually tagged with 14 part-of-speech tags (Appendix A). An example of a token sequence taken from the file is:
"- **F** infektion **N** med **S** enterohemorragisk **A** E.coli **N** (**F** EHEC **N**) **F** . . . ".

5.3.2 Tilburg Memory-Based Learner

Tilburg Memory-Based Learner (TiMBL)[13] is a program that implements several memory-based learning algorithms and metrics (Daelemans et al. 2004). The program is optimized for fast classification by using several indexing techniques and heuristic approximations (such as IGTREE and TRIBL).

One way of using TiMBL is to apply a learning method to a dataset, and analyze the output from the program to extract information about the data. The dataset for training and testing consists of descriptions of instances (one instance per line) in terms of a fixed number of feature-values (these are presented in Subsection 6.3.1). A particular problem can be tackled using alternative methods and the results evaluated, a flexibility which makes the program suitable for selecting the most appropriate method for a certain learning task.

The aim of using TiMBL is to train a classifier which can predict the class of new, previously unseen pairs as correctly as possible.

[13] http://ilk.uvt.nl

❖ 6

System overview

This chapter describes the design and implementation of the acronym recognition system. It starts with an overview of the system and describes the steps that are used to deal with the acronym acquisition problem.[14] Section 6.2 continues with a detailed description of the identification process. In Section 6.3 the choice of features and machine learning algorithm is justified.

6.1 System architecture

The system described here divides the acronym acquisition problem in to three steps:
Step (1): Acronym identification. The first step is to search for an acronym candidate (a single token). This is done by weak heuristics which reduce the number of candidates.
Step (2): Definition identification. **(2a)** When an acronym candidate is found, the search space for the definition candidate is determined according to the acronym length and the line[15] where it appears. **(2b)** Part-of-speech tags and non-alphanumeric characters are used to reduce the string and identify a definition candidate (a sequence of one or more tokens).
Step (3): Acronym-definition classification. As a result of the identification process (step 1 and 2), a list of acronym-definition candidates is generated. Each acronym candidate (AC) and definition candidate (DC) should then be classified (labelled) as either positive or negative.[16] This is done according to the following rules:
(3a) if AC = A then AC is labelled as positive, otherwise (AC,DC) is labelled as nega-

[14] *Acronym acquisition* is a lexicon-building process in which acronyms are collected either manually or automatically from textual evidence.
[15] Defining "line" as: a sequence of tokens which ends with a full stop.
[16] A "positive" acronym-definition candidate is considered to be a "correct" acronym-definition pair by humans, defines as (A, D).
A "negative" acronym-definition candidate is not considered to be an acronym-definition pair by humans.

tive.
(3b) if AC is positive and DC = D then (AC,DC) is labelled as positive, otherwise (AC,DC) is labelled as negative.

Steps 3a and 3b are the steps we want our classifier to learn. A task which requires training. Figure 6.1 illustrates the training phase (phase 1), in which manual modification and correction of the data yields a set of labelled (AC,DC). This set (presented as feature vectors) is used as input data to train and test the machine-learning algorithm.

In this phase, the identified candidates and the originating line are presented on an html page (html results are viewed in Diagrams 7.1 and 7.2) which allows modifications, such as reducing a definition string, marking a candidate pair as invalid etc. Manual classification adds an additional classification possibility to steps 3a and 3b:
(3c) if AC is positive, modify DC so that $DC \Rightarrow D$, after which (AC,DC) is labelled as positive.

In phase 2, illustrated in Figure 6.2, the trained classifier is applied to the extracted pairs (AC,DC) presented as feature vectors. Classification, i.e. steps 3a and 3b, is done automatically.

As a result of a successful classification process acronym-definition pairs labelled positive are added to the database. The information stored in the database is the acronym-definition pair and its feature vector. Each acronym-definition in the database is assigned an unique identifier (ID), to enable fast access to the information. Additional information stored in the database is the acronym pair frequency.[17]

There are two rules for adding an acronym-definition pair (A,D) to the database (DB):
(1) If (A,D) exists in DB, increase frequency with 1.
(2) If (A,D) does not exist in DB, add (A,D), set frequency = 1.

6.2 Acronym-definition identification

After the input text is POS-tagged, it is passed through a set of heuristics for determining which of the tokens in each line are acronym candidates and definition candidates.

[17] Information about how frequent is the acronym-definition pair is necessary for counting the probability of each pair. This information will be used in later development, when dealing with acronym disambiguation.

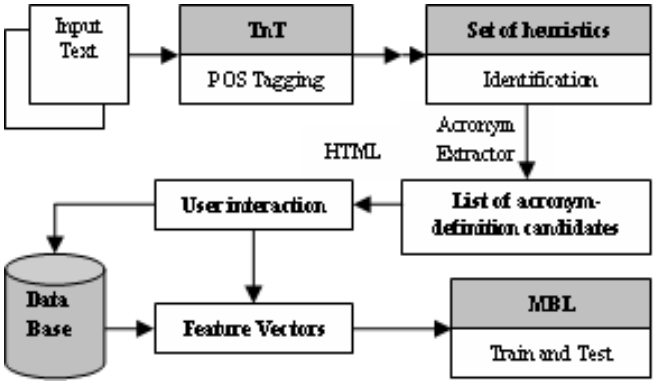

Fig. 6.1. System architecture, phase 1.

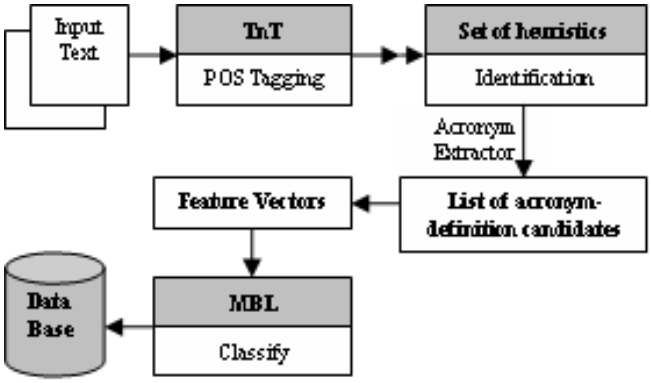

Fig. 6.2. System architecture, phase 2.

6.2.1 Heuristics for acronym candidates

An acronym string (A) is accepted as a valid candidate if the following conditions are satisfied:

(1) PosTag(A) is either N, Y or X, noun, abbreviation or foreign word. In case X, it must consist of at least 2 upper-case letters.

(2) (A) is not in the list of noise words, nor names.

The list of noise words contains words such as “by”, “cm”, “ml”. The list of names includes person names as “The-Hung Bui”, “Hans”.

(3) (A) does not contain characters such as ’⟨’, ’⟩’, ’(’, ’]’, ’=’.

(4) $2 \leq |A| < 14$.

$|A|$ is the length of the acronym string which refers to the number of characters in the string including digits and punctuation characters.

6.2.2 Heuristics for definition candidates

One of the main challenges of the acronym acquisition task is to select the optimal definition candidate. The majority of errors in text-based methods relate to the size of the window used when searching for potential definitions. In this approach, additional pre-processing steps are used to recognize possible definition candidates.

The initial search space for a definition string is $|A|$*4 (words). The string is reduced according to the following:
(1) The string contains a bracket, colon, semi-colon, question mark or exclamation mark.
(2) The maximum length of the string is min($|A|$ + 5, $|A|$*2) (Byrd and Park 2001). Definition length is measured by number of words, not including prepositions and determiners. Acronym length is measured by number of characters.
(3) The string includes characters such as: ')', ']', ', ', that indicates the end of the definition.
(4) The part of speech tags of the words in the string, i.e. the string cannot start and end with a verb.[18]

This general identification process is done according to non-alphanumeric characters the appearance of the acronym i.e. if the acronym is found within parentheses, the definition is likely to appear before. If the acronym is followed by a '=' the definition is likely to appear after the acronym. To illustrate the process, consider the following sentence:

"endoplasmatiskt retikel sER och rER , cisternae Living
Donor Liver Transplantation (LDLT), idag . . . ".

A potential definition string is "sER och rER , cisternae Living Donor Liver Transplantation". However the string is reduced to: "cisternae Living Donor Liver Transplantation", since the word "cisternae" appears after a comma. The non-alphanumeric characters used by the algorithm during the identification process are shown in Table 5.1.

When the identification process is completed, the acronym-definition candidates are submitted to a sequence of tests to remove wrong acronym-definition pairs. Invalid pairs are pairs for which their definition string:
(1) does not include letters, or (2) contains only one string that includes '@', 'www', or (3) has the same length as the acronym string, or (4) consists of only upper case letters.

[18] This strategy was inspired by Pustejovsky and colleagues that determine the definition window size by morpho-syntactic properties where only noun phrases are considered as possible candidates.

6.3 Supervised machine-learning

In order to apply supervised learning algorithms, acronym-definition candidates must be represented as feature vectors. Selection of relevant features is crucial to the learning process as well as the algorithms and methods which are chosen to train the classifier.

6.3.1 Choice of features

The feature subset was selected in two steps. First, independent, general characteristics of the data were chosen. Second the set that produced the most promising subset was filtered. The former was achieved by simply looking at negative and positive examples and observing their differences. The latter was achieved by ignoring certain features during the training of the machine-learning algorithm and observing how these ignored features affected the results.

This selection process resulted in a set of 14 features (Appendix B.3) used to describe each acronym-definition pair. The 15th feature is the class to predict, i.e. positive(+) or negative(-) candidate.

The choice of some features was inspired by previous authors (Chang et al. 2002; Nadeau and Turney 2005). Some features resulted from machine-learning evaluation, showing that certain features contributed more to correct classification than others. These features are:
Feature 2: Whether the acronym appears before the definition.
Feature 3: The distance (in words) between the acronym and the definition.
Feature 4: Number of participating letters matching the first letter of a definition word.
Feature 5: Number of characters in the acronym.
Ignoring these features decreased the performance drastically.

The remaining features were chosen mainly because the system does not employ letter or pattern matching. These features are based on common characteristics that describe acronym variations and are normally used (by pattern-based systems) to identify and match an acronym to its definition (see Table 4.1).

The example below shows how an acronym-definition pair is presented to the machine-learning algorithm according to this set of features:
The acronym-definition pair *icke-typbara H. influenzae (NTHi)*, that appears in line: “De bakterier som användes var icke-typbara H. influenzae (NTHi) ,”, is presented as the feature vector: 1,0,0,2,1,3,1,0,4,26,3,1,1,3,+

6.3.2 Choice of algorithm

In previous experiment (Dannélls 2006) a number of algorithms were tested on a single train/test partition (train 80%, test 20%). Such a test does not factor out the bias of choosing this particular partition. When Daelemans et al. (2004) present their experiment (Daelemans, Chapter 4), they point out that a more reliable generalization accuracy measurement is used in real experiments, e.g. 10-fold cross-validation. This means that 10 separate experiments are performed, and in each run 90% of the data is used for training and 10% for testing, in such a way that each instance is used as a test item exactly once.

Another reliable way of testing the real error of a classifier is leave-one-out. In a leave-one-out test, every data item in turn is selected once as a test item, and the classifier is trained on all remaining items. Accuracy of the classifier is the number of data items correctly predicted. Using this option no test file is provided, testing is done on each pattern of the training file, by treating each pattern of the training file in turn as a test case and the whole remainder of the file as training cases.

The algorithm used in the machine-learning experiment is IB1, since it is the only algorithm which is combined with the leave-one-out test.

Distance metrics and Feature weighting
Distance metrics are used for influencing the definition of similarity. The metrics provided in TiMBL are: Overlap, MVDM, Jeffrey divergence, Dot product and Numeric.
These five were used to create models and were evaluated. Four of these gave outstanding results, these are: Overlap, MVDM, Dot product and Numeric. Their results are presented in Table 7.1

Distance metrics are combined with several feature weighting methods. Feature weighting is used to measure how much information each feature contributes to the knowledge of the correct class label. The weighting methods provided in TiMBL are: No weights, Gain Ratio, Information Gain Chi-square and Shared variance. In the experiment all of these methods were tested, however the best results attained in the combination: No weights (weight = 1), Gain Ratio (weight = normalize information for features with different numbers of values), and Information Gain (weight = how much information each feature contributes).

When using different weights it is useful to extrapolate from several most similar examples in the memory which is done by experimenting with different values of

k-nearest neighbour.

k-nearest neighbour

A memory-based learner learns from a set of instances stored in a database where each instance is tagged with a particular category. The instances are taken from a manually annotated training set. To classify new instances, the system searches the database for the instance or a set of instances that is most similar to the new instance. In other words, the system searches for the nearest neighbour of the new instance. The technique descends from the k-nearest neighbour technique, and only the k nearest neighbours are considered. Often k is taken as 1 but it is one of the objectives of experiments with memory-based methods to find an optimal value for k.

Class voting

Another choice to be made is class voting which is a about how much influence each neighbour has. This option is used for extrapolation from the nearest neighbour set. The classes available in TiMBL are: Majority voting, Inverse Distance weighing, Inverse Linear weighing, and Exponential Decay weighing. These options where tested with k=3, the results are given in Table 6.1.

Class voting method	Accuracy
Majority voting	98.6 %
Inverse Distance	98 %
Inverse Linear	97 %
Exponential Decay	97.7 %

Table 6.1. Class voting methods results.

As table 6.1 shows Majority voting is most accurate. It is also the most straightforward method for letting the k nearest neighbours vote on the class of a new case (Dealmans subsection 5.1.7) where each neighbour receives equal weight. Thus, in case of a tie the class with the highest number of votes is chosen. It is therefore important to use this function with an odd value for k. For this reason the values of k-nearest neighbour tested are: k=1, k=3, k=5, k=9.

7

Evaluation and results

This chapter evaluates the performance of the system, and presents the results obtained with the machine-learning algorithm.

7.1 Part-of-speech tagger

The tagger was trained on a dataset which consists of 14,778 tokens. The trained model was tested against a manually tagged set consisting of 4,823 tokens. The results showed that the tagger successfully tagged 91.8% of the given tokens, i.e. 4,428 tokens were assigned a correct tag. The tagger failed with 395 tokens out of 4,823.

This tagger trained in this way was used in the system to tag the input text file as the first stage in the recognition process.

7.2 Identification process

The un-annotated version of the dataset (described in Section 5.2) was used as the input file to test and evaluate the identification process. This set contains 861 acronym-definition pairs.

The heuristics presented in Section 6.2 enable 898 acronym-definition candidates to be extracted. Evaluation of the acronyms found (without their definition) showed that 845 correct acronyms were extracted, yielding 98% recall at 94% precision. Thus, only 16 acronyms were missed. Examples of some acronyms that were not identified, and reasons why they were omitted from the final acronym list are as follows:
(1) Acronyms consisting of more than one token i.e. 'PUU N', 'P S A'.
(2) Acronyms removed due to their definition string that might have included symbols and letters such as '@', 'www'.
(3) Wrong interpretation by the tagger.

Evaluation of the correctly identified acronym-definition pairs showed 65% recall, at 62% precision. There were 556 exact matches of correctly matched definition strings.

It is important to note that additional characters such as space, comma, asterisk, etc. appearing at the beginning and/or end in the definition strings caused incorrect matching during evaluation, for example the definition “anti-cyclic citrullinated peptide , ”, was evaluated as a negative example, even though the correct definition is: “anti-cyclic citrullinated peptide”.

Most mistakes were made due to boundary errors, i.e. the extracted definition strings contained words such as “och”, “on”, “and”, and other words, mostly nouns, which were not included in the correct definition. For example:
Found: *sjukdomen multipel skleros (ms)*
Correct pair is: *multipel skleros (ms)*,
Found: *de s å kallade apolipoproteinerna , apo*
Correct pair is: *apolipoproteinerna , apo* .

Mistakes of this type result from avoiding pattern-based matching. This was an intentional design decision to widen the variety of acronym-definition pairs which could be extracted.

An additional reason for wrong definition matching is the fact that some lines contain an acronym without its definition. Examples of such cases are:
1) “. . . Inegy är indicerat som tilläggsterapi till diet hos patienter med HoFH.”, no definition is given for the acronym ’HoFH’; 2) “the presenting symptom in 10-15 % of patients with CF”, no definition is given for the acronym ’CT’.

Examples of pairs that were found successfully, and which are normally hard to find using common pattern-based methods are: *allmän hälsa (GH)*; *kvalitetsjusterat levnads år (QALY)*; kammarflimmer (VF).

There are also a few cases where a definition string was extracted from the preceding line instead of the following line, e.g. in the following line:
“ histon-acetyltransferas , HAT , respektive histon-deacetylas ,”.
The acronym ’HAT’ was matched with the definition string: “histon-deacetylas” instead of with the definition string: “histon-acetyltransferas”.

anti-mikrobiologiska medel och kemoterapi (ICAAC) .
Line: 3
" Losartan Intervention For Endpoint reduction in hypertension study (LIFE) " är en omfattande studie (9 193 patienter) som jämför behandling baserad på AT1-receptorblockeraren losartan (Cozaar) eller beta-1-receptorblockeraren atenolol (Atenolol Nordic rekommenderas i baslistan) .
Line: 4
" Normalt " när RNA för över information från den dubbelsträngade DNA-genen till proteinet har det en enkelsträngad form , kallad mRNA (budbärarRNA) .
Line: 5
* endoplasmatiskt retikel - sER och rER , cisternae * Living Donor Liver Transplantation (LDLT) , i dag huvudsakligen högerlobsdonation , var utöver föredrag , posters och State-of-the-Art-Lecture föremål för en hel eftermiddags diskussioner under Transplantationskursen .
Line: 6
* Medicinsk utrustning , inklusive MR utrustning (magnetresonanstomografi) , terapeutisk strålning (såsom cancerstrålningsbehandling) och TENS (transkutan elektrisk nervstimulering) COPEC (Council for Physical Education for Children) of the NASPE (National Association for Sports and Education) an association of the AAHPERD (American Alliance for Health , Physical Education , Recreation and Dance) .
Line: 7
OT = ortodrom takykardi vid WPW-syndrom , AVNRT = AV-nodal reentrytakykardi , LQTS = Lång QT-syndrom , VT = ventrikeltakykardi , SVT = supraventrikulär takykardi , VOC = organiskt hjärtfel (vitium organicum cordis) ACE - Angiotensin Converting Enzyme , ett enzym som förändrar angiotensin 1 till angiotensin II .
Line: 8
CRF (Case Report Form) kallas läkarens blankett som fylls i för hand med grunddata för varje försöksperson , till exempel längd , vikt , födelsedatum och liknande .
Line: 9
TUPP - Tidig Upptäck av läkemedelsrelaterade Problem och Påverkan på läkemedelsförskrivningen Fédération Internationale Pharmaceutique , FIP , och den internationella läkarorganisationen World Medical Association , WMA , har inlett en dialog för att definiera de båda professionernas respektive roller inom hälso- och sjukvården .
Line: 10
Functional Independence Measure (FIM) samt Instrumental Activity Measure (IAM) för bedömning av ADL-funktion .
Line: 11
Gastroesophageal reflux (GERD) var vanligare bland patienter i peritoneal dialys (PD) i jämförelse med patienter i hemodialys och predialytiska patienter med kronisk njursjukdom .
Line: 12
Gastrointestinal Symptom Rating Scale (GSRS) och Visceral Sensitivity Index (VSI) , livskvalitet ; Hierarchical cluster analysis (HCA) and multidimensional scaling (MDS) were performed using STATISTICA 6 software (StatSoft Inc. , Tulsa , OK) or TIGR MeV Multi Experiment Viewer .
Line: 13
Implanterbar defibrillator (ICD) består av ettpacemakerliknande system , vars generator harförmåga att avge bifasiska chocker vid VT eller kammarflimmer (VF) .
Line: 14
Long-term oxygen therapy (LTOT) prolongs survival when given to patients with severe hypoxia due to chronic obstructive lung disease (COPD) .
Line: 15
Svensk förening för Endoskopi och Gastroenterologi Personal (SEGP) hade sitt program och samtidigt hade Svensk Gastroenterologisk Förening (SGF) sitt .
Line: 16
The Journal of the American Medical Association (JAMA) , och The New England Journal of Medicine (NEJM) .
Line: 17
Akut intermittent porfyri , AIP , är en ärftlig bristsjukdom .
Line: 18
American College of Sports Medicine (ACSM) publicerade den första versionen rörande konditionsträning 1978 .
Line: 19

Fig. 7.1. Example of the annotated output file.

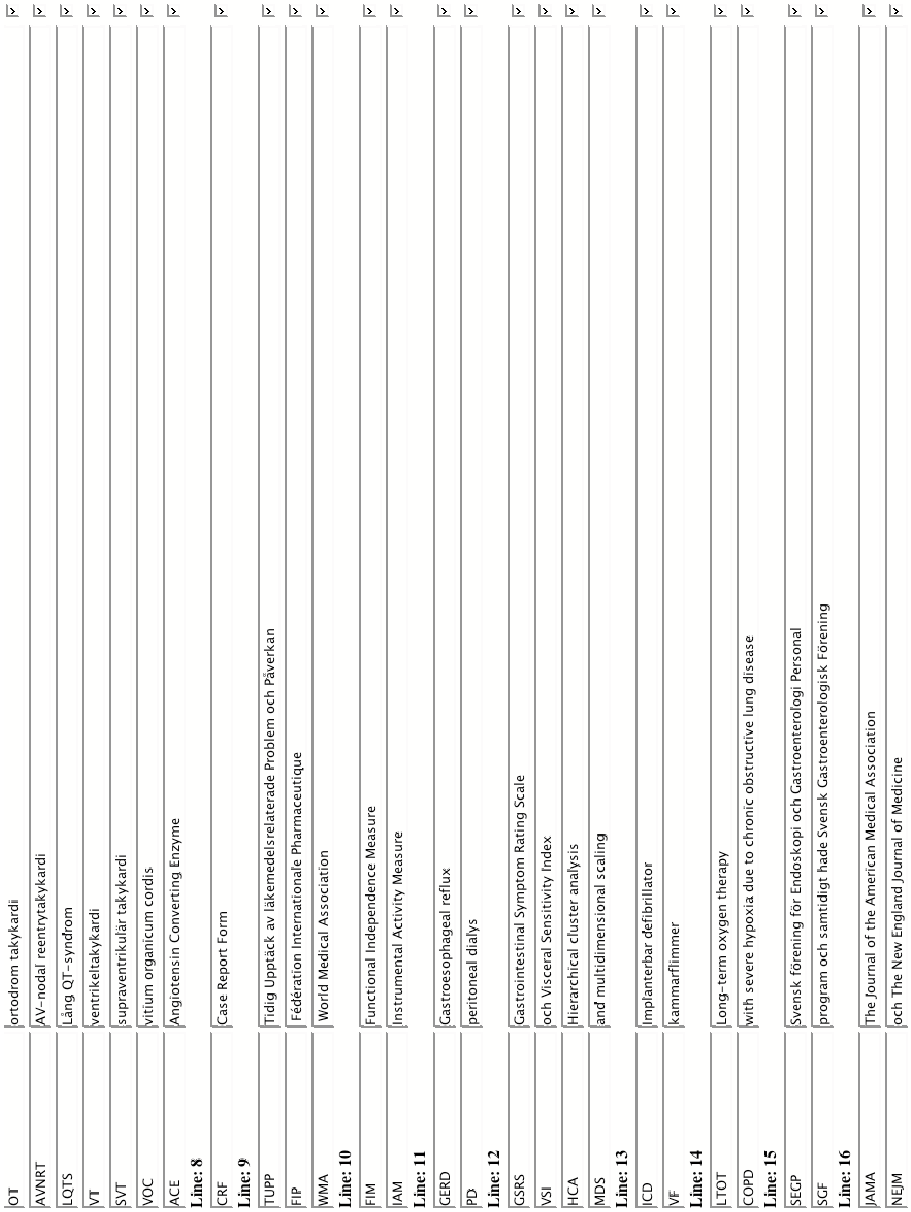

Fig. 7.2. Example of the identified acronym-definition candidates.

7.3 Machine-learning algorithm

To train and test different input possibilities in TiMBL (described in Subsection 6.3.2), a set of 1060 acronym-definition pairs were presented as feature vectors, 681 of these were positive examples and the remaining, 379 examples were negative. The features used to present the pairs are specified in Appendix B.3 and are discussed in the Subsection 6.3.1.

We experimented with different distance metrics, weighting possibilities and k nearest neighbour values, using TiMBL. The results are shown in Table 7.1. The machine-learning experiment shows that the IB1 algorithm is appropriate for the task. The results shows (see Table 7.1) that between 93% to 99% candidates were correctly classified when testing different modelling combinations using this algorithm.

A closer look at the results given by testing different distance metrics, weighting possibilities and k nearest neighbour values shows that the Overlap Metric gives lower precision with higher k values, i.e. 3, 5 and 9. This metric performs best with k = 1. The Dot-Product Metric seems to give best balance between precision and recall when weighting = 1. MVDM give rather good balance between precision and recall in most cases.

The Numeric Metric seems to give the best balance through out the table. A good combination seems to be with Information Gain weighting and k=3. Examination of classification errors found when testing this model showed that many errors were caused by mismatch of words like “the”, “of”, for example the algorithm found: “Journal of the American Medical Association” instead of “The Journal of the American Medical Association”.

k	Weight		Overlap	MVDM	Dot-product	Numeric
1	No	P	98.4 %	98.8 %	97.7 %	98.5 %
		R	98.3 %	98.8 %	98.5 %	98.5 %
1	GR	P	98.4 %	98.4 %	95.5 %	99.1 %
		R	97.5 %	98.7 %	99.2 %	98.7 %
1	IG	P	97.9 %	98 %	94.6 %	98.7 %
		R	97 %	98.3 %	99.2 %	99.1 %
3	No	P	94 %	98.5 %	98.3 %	98 %
		R	98.2 %	98.2 %	98.3 %	98.9 %
3	GR	P	98.1 %	98.4 %	95.2 %	98.4 %
		R	98.8 %	98.5 %	99.1 %	98.8 %
3	IG	P	97.3 %	98.2 %	93.4 %	**98.8** %
		R	98.5 %	98.2 %	99.5 %	**98.8** %
5	No	P	92.4 %	98.5 %	98.6 %	98.2 %
		R	98.8 %	98.2 %	98.3 %	98.5 %
5	GR	P	97.8 %	98.5 %	96.1 %	98.2 %
		R	98.7 %	98.7 %	99.1 %	98.8 %
5	IG	P	98 %	98.5 %	93.4 %	98.5 %
		R	98.3 %	96 %	98.8 %	98.7 %
9	No	P	93 %	98.4 %	98.8 %	98.8 %
		R	98.8 %	98.2 %	97.5 %	98.6 %
9	GR	P	98 %	98.3 %	95 %	98.7 %
		R	98 %	98.1 %	98.8 %	98.6 %
9	IG	P	97.1 %	98 %	93.6 %	98.5 %
		R	98.9 %	98.2 %	98.8 %	98.5 %

Table 7.1. ML algorithm results using leave-one-out test.

No = No weighting, GR = Gain Ratio, IG = Infromation Gain,
P = Precision, R = Recall.

❖ 8

Discussion

Examination of previous work (Chapter 3) showed that using multiple alignments of text (Taghva and Gilbreth 1999; Schwartz and Hearst 2003) to map acronyms to their definitions is inappropriate strategy since: (1) an acronym can contain letters that do not occur in its definition; (2) a definition can include letters that do not correspond to any of the letters in the acronym.

Larkey (2000) showed that accepting a wide range of acronym patterns results in high recall but limits precision, e.g. the results of the canonical/contextual algorithm, Table 3.1, which allows a large variety of acronym pairs. By contrast a limited set of patterns decreases recall but gives very good precision.

Pustejovsky et al. (2001) showed that wide range of acronyms can be recognized if the task of determining the search space for the definition, and the task of matching acronym-definition are treated separately. Furthermore their implementation demonstrated the advantage of using part-of-speech information. Such information allows recognition systems to set a natural boundaries for definition strings. There are two reasons why their system (AcroMed) attained high accuracy: (1) syntactic information is used for determining the search space for the definition; (2) regular expressions are used for the acronym-definition matching.

The results reported by Adar using his algorithm (SaRAD) on the same dataset used in AcroMed are very impressive but not directly comparable to the results reported by Pustejovsky et al.[19].

The method presented in this paper is inspired by the work of Pustejovsky et al. It confirms the advantage of using syntactic information, as well as the advantages of

[19] Adar experimented with a threshold = 0, while Pustejovsky et al. experimented with a threshold = 1.5.

separating the acronym-definition recognition process in two separated steps.

An additional advantage shown in this thesis is the ability to recognize complex acronym patterns by avoiding matching rules and other letter alignments methods.

Nadeau and Turney (2005) use a large set of features to describe the acronym-definition pairs. They explain the significance of each feature, and point out that not all the features they have considered need to be used in conjunction.

Feature selection is crucial to the performance of the machine-learning algorithm, and it is of great importance to test the relevance of each feature employed. In this work the contribution of each feature (Subsection 6.3.1) was evaluated (using Gain Ratio weighting). The results indeed showed that certain features contribute more than others to decision-making during the learning process.

The significance of the each feature varies from one acronym to another. Some features show good discrimination with some examples in the test set, while other features are more useful for others. It is therefore important that the set of features spans the required classification space. Which was one reason for employing rather large number, and representative set of features in this work.

In earlier experiments, manual partition of the data (train and test files) was used in the machine-learning experiment. Such tests do not eliminate the bias of choosing a particular partition, and should be avoided. In this work leave-one-out test is performed on the whole data. Different weights, and values of k-nearest neighbour were tested.

Some conclusion could be drawn from the machine-learning results presented in Table 7.1, but it was not clear which of the tested combinations are most suitable for this task. These machine-learning results can be explained by the fact that the experiment was based on a dataset containing only 2% negative examples. To train a machine-learning algorithm to classify reliably, one should have equal numbers of negative and positive examples in the dataset.

❖ 9

Conclusions

This thesis has been concerned with the task of recognizing acronyms and their definitions in Swedish biomedical texts. A system to deal with the acronym recognition task has been presented. Here we attempt to summarise the capabilities and the limitations of this system, and present the achievements of this work.

The system utilizes: (1) a set of heuristics to identify and extract acronym-definition candidates; (2) a memory-based classifier that can be trained to predict the class of the acronym-definition pairs.

The general heuristics used for identifying acronym-definition candidates allow to extract significant number of acronym-definition pairs. This design makes the algorithm flexible and allows it to recognize a wide variety of acronym-definition patterns.

The identified pair candidates can be used as training data for the machine-learning algorithm to test and compare different modelling combinations. This architecture (viewed in Figure 6.1) makes it possible to train a machine-learning classifier continuously on several texts and languages.

For training purposes, the system relies on manual correction to classify the extracted pair candidates. The trained classifier can then be applied directly to the extracted acronym-definition candidates. These will be labelled automatically and added to the database (see Figure 6.2). This is a result of a successful classification process.

The attempt to train a classifier on the available dataset by testing different modelling combinations (Table 7.1) showed it is not easy to choose parameters which will give the best-results. Numeric Metric in combination with Information Gain weighting and k=3 seems to be a good choice, and deserves further training and testing in the future.

In general, further training and testing with different models are required. The algorithm should be trained on a set of instances (vectors) containing equal numbers of positive and negative examples to confirm the results presented in this experiment, and maybe even improve performance.

In summary, training different classifiers on the examples that were available for this experiment yielded promising results, and I believe this approach will allow non-trivial prediction on data where unseen acronym-pairs are presented for the first time.

9.1 System capabilities

The system does not require an annotated input file to classify the extracted pairs in order to train and test the supervised machine-learning algorithm. If such file is not available, results are presented to the user for manual classification and correction. It is designed for use by lexicographers, and skilled people who can reliably classify the identified acronym-definition pairs.

The acronym database is being constantly updated while the system is running. It enables retrieval and analysis of the stored information.The information stored in the database will subsequently be accessible by electronic search engines.

The identification process is fast and suitable for different input data. One of the system advantages is that it presents the whole line where the acronym-definition pairs were found, allowing the user to detect where mistakes occur and correct them easily. This allows fast modification and corrections.

The system allows testing different input settings with TiMBL, e.g. different feature weighting schemes and different k values – an advantage that makes the system attractive and useful.

An additional facility that is implemented in the system is the possibility to evaluate the identification result. In order to use this facility the user must provide a list of correct acronym-definition pairs that exists in the input file from which acronyms will be extracted. This file must consist of two tab separated fields in each line. The first field contains an acronym string with its definition string in the corresponding second field.

9.2 System limitations

Wrong interpretation by the tagger limits system performance, and further training is needed to achieve satisfactory results (the tagging results showed 91.8%, Section 7.1). The tagger is trained on a dataset containing only 14,778 tokens, and is evaluated on a small dataset, consisting of 4,823 tokens. For more reliable evaluation, the tagger should be trained on a larger dataset.

The tag-set is very small. Only 14 tags are used, for example proper nouns and nouns are tagged with N. In its present implementation, the system uses a list of pre-defined names, which is a big limitation. Such a list cannot possibly include all the names to be found in different texts and may cause low precision. A suitable tag for proper nouns should be PN. The abbreviation tag Y is confusing since only certain acronyms are tagged with this tag. Preferably, all acronyms should be interpreted as nouns N.

The algorithm used in the system does not cover acronyms where digits are used to indicate repetitive letters in the expansion, such as ’W3C’ for “World Wide Web Consortium”, or for symbolic matching, such as ’C’ instead of “see” or ’2’ instead of “to”, nor for acronyms that include space characters.

9.3 Research contribution

One of the advantages of this approach compared with other known methods is that it handles complex acronym structures in Swedish. As shown in Section 7.2, infrequent acronym structures are found and extracted. It is a promising approach, allowing a wide variety of acronyms to be detected.

The main advantage of the system is that it allows different algorithms and modelling combinations to be trained, tested and compared on several acronym texts.

The most serious problem with supervised classification systems is that a large training set is essential for good performance. Building a training set by human labeling is time consuming, labour intensive and expensive. Our system is structured in such way that the tagged output file can easily be saved which will lead to an automatically growing dataset. Such a design will surely be useful for many NLP tasks and research purposes.

❖ 10

Future work

There are several extensions and modifications that could be carried out to improve the system. This chapter suggests some of them.

The current tagger limits the performance of the acronym recognition system. The tagger should therefore be trained on a larger dataset and a larger tag set.

The system structure allows extension, so new facilities can readily be added. One such facility is direct interaction with sites available on the World Wide Web. Another facility is creating a large dataset which contains tagged Swedish acronym-definition pairs. One way of generating a large acronym-definition dataset is by saving the tagged data files generated by the system during system usage. These files include the input text and the corrected pairs after manual modification.

For testing purposes, it is sufficient to include some negative examples in the training data. For larger tasks, such as training a model for classification of new (unseen) instances, the training set should include similar numbers of negative and positive instances. More accurate evaluation when training the machine learning classifier requires a file for random access of negative examples to balance the number of positive examples.

Negative acronym-definition pairs must also be represented. These could be extracted randomly from the examples needed for the machine-learning experiment (and equal to the number of positive instances). One simple possibility is to save the negative acronym-definition pairs and their vectors in a text file or in a (database) table accessible to the system. Such a facility is needed for accurate and reliable training and testing results.

Work is still required to test the robustness of the system. We need to investigate

its ability to adapt to Swedish texts with different structure from the texts used during system development. It would also interesting to investigate how well the system performs on texts written in Dutch, Finnish, etc. and test Zahariev's assumption (mentioned in Section 4.3).

Maximum Entropy (ME) modelling is a statistical technique that has been used successfully in recent years for NLP tasks such as pos-tagging and boundary detection. This technique will be used in the future to disambiguate the meaning of acronyms stored in the database. The technique was suggested by Pustejovsky (Pustejovsky et al. 2001) and I believe it could help to find the correct interpretation of a given acronym.

Bibliography

[Adar 2002] Adar, E. (2002, September). Sarad: A simple and robust abbreviation dictionary. Technical report, HP Laboratories. http://www.hpl.hp.com/research/idl/projects/abbrev.html.

[Brants 2000] Brants, T. (2000). Tnt - a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference*, pp. 224–231.

[Chang et al. 2002] Chang, J. T., H. Schütze, and R. B. Altman (2002). Creating an online dictionary of abbreviations from medline. *Journal of American Medical Informatics Association (JAMIA) 9*(6), 612–620. http://abbreviation.stanford.edu/.

[Cheng 1994] Cheng, T. O. (1994). Acronymophilia: the exponential growth of the use of acronyms should be resisted. *BMJ 309*, 683–4.

[Cormen et al. 1990] Cormen, T., C. Leiserson, and R. Rivest (1990). Introduction to algorithms. MIT Press, Cam bridge, MA.

[Daelemans et al. 2004] Daelemans, W., J. Zavrel, K. van der Sloot, and A. van den Bosch (2004, December). Timbl: Tilburg memory-based learner. Technical Report version 5.0.1, Computational Linguistics Tilburg University. ILK Technical Report - ILK 04-02.

[Dannélls 2006] Dannélls, D. (2006). Automatic acronym recognition. In *Proceedings of the 11th conference on European chapter of the Association for Computational Linguistics*, pp. 167–170. Trento, Italy.

[Kokkinakis 2006] Kokkinakis, D. (2006). Collection, encoding and linguistic processing of a swedish medical corpus: The medlex experience. In *Proceedings of the 5th Language Resources and Evaluation Conference*. Genoa, Italy.

[Larkey et al. 2000] Larkey, L. S., P. Ogilvie, A. M. Price, and B. Tamilio (2000). Acrophile: An automated acronym extractor and server. In *Proceedings of the ACM Digital Libraries conference*, pp. 205–214. University of Massachusetts, Dallas TX. http://ciir.cs.umass.edu/ciirdemo/acronym/.

[Mitchell 1997] Mitchell, T. M. (1997). *Machine Learning*. MIT Press and The McGraw-Hill Companies,Inc. Singapore.

[Nadeau and Turney 2005] Nadeau, D. and P. Turney (2005). A supervised learning approach to acronym identification. In *Proceedings 18th Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 319–329. Victoria, BC, Canada.

[Park and Byrd 2001] Park, Y. and R. J. Byrd (2001). Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*. Pittsburgh, PA.

[Pustejovsky et al. 2001] Pustejovsky, J., J. Castaaño, B. Cochran, M. Kotecki, M. Morrell, and A. Rumshisky (2001). Linguistic knowledge extraction from medline: Automatic construction of an acronym database. In *10th World Congress on Health and Medical Informatics (Medinfo 2001)*. http://medstract.med.tufts.edu/acro1.1/index.htm.

[Schwartz and Hearst 2003] Schwartz, A. S. and M. A. Hearst (2003). A simple algorithm for identifying abbreviation definitions in biomedical texts. In *Proceedings of the Pacific Symposium on Biocomputing (PSB)*. University of California, Berkeley. http://biotext.berkeley.edu/software.html.

[Simpson and C.Weiner 1989] Simpson, J. A. and E. S. C.Weiner (1989). Oxford english dictionary. Clarendon Press, Second edition.

[Spasic et al. 2005] Spasic, I., S. Ananiadou, J. McNaught, and A. Kumar (2005). Text mining and ontologies in biomedicine: Making sense of raw text. *Oxford Journals 6*(3), 239–251. Briefings in Bioinformatics.

[Taghva and Gilbreth 1999] Taghva, K. and J. Gilbreth (1999). Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition (IJDAR) 1*, 191–198. http://www.acronymfinder.com/.

[Yeates 1999] Yeates, S. (1999). Automatic extraction of acronyms from text. In *Proceedings of the New Zealand Computer Science Research Students' Conference*, pp. 117–124. University of Waikato, Hamilton, New Zealand.

[Yeates et al. 2000] Yeates, S., D. Bainbridge, and I. H. Witten (2000). Using compression to identify acronyms in text. In *Data Compression Conference*, pp. 582. http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:cs/0007003.

[Yu et al. 2001] Yu, H., M. G. Hripcsak, and C. Friedman (2001). Mapping abbreviations to full forms in biomedical articles. *Department of Medical Informatics 9*(3), 262–272. Columbia University, New York, USA.

[Zahariev 2004] Zahariev, M. (2004). *A Acronym*. Ph. D. thesis, SIMON FRASER UNIVERSITY.

❖ A

Appendix: The tag-set

A adjective, participle
C conjunction/subjunction
D determiner, article
F punctuation mark
I interjection
Mnumeral
N noun (including proper nouns)
P pronoun
R adverb
S preposition
V verb
X foreign word (for example, an English word)
Y abbreviation
Z infinitive marker (swe. ”att” eng. ”to”)

❖ B

Appendix: Features for supervised learning

B.1 Chang et al., Online dictionary of abbreviations

1. Percentage of letters in abbreviation in lower-case;
2. Percentage of letters aligned in the beginning of a word;
3. Percentage of letters aligned in the end of a word;
4. Percentage of letters aligned on a syllable boundary;
5. Percentage of letters aligned immediately after another letter;
6. Percentage of letters in the abbreviation that are aligned;
7. Number of words in the prefix not aligned to the abbreviation;
8. Average number of aligned letters per word;
9. Normalization constant.

B.2 Nadeau and Turney, Supervised learning approach

1. The number of participating letters matching the first letter of a definition word;
2. (1) Normalized by the acronym length;
3. The number of participating definition letters that are capitalized;
4. (3) Normalized by the acronym length;
5. The length (in words) of the definition;
6. The distance (in words) between the acronym and the definition;
7. The number of definition words that do not participate;
8. (7) Normalized by the definition length;
9. The mean size of words in the definition that do not participate;
10. Whether the first definition word is a preposition, a conjunction or a determiner;
11. Whether the last definition word is a preposition, a conjunction or a determiner;
12. Number of prepositions, conjunctions and determiners in the definition;
13. Maximum number of letters that participate in a single definition word;
14. Number of acronym letters that do not participate;

15. Number of acronym digits and punctuations that do not participate;
16. Whether the acronym or the definition is between parentheses;
17. The number of verbs in the definition.

B.3 Dannélls, Acronym recognition

1. Whether the acronym or the definition is between parentheses;
2. Whether the acronym appears before the definition;
3. The distance (in words) between the acronym and the definition;
4. Number of participating letters matching the first letter of a definition word;
5. Number of participating definition letters that are capitalized;
6. Number of acronym letters that participate;
7. Number of letters aligned on a punctuation boundary;
8. Number of letters aligned immediately after another letter;
9. Number of characters in the acronym;
10. Number of characters in the definition;
11. Number of words in the definition;
12. Number of acronym letters that do not participate;
13. Number of definition letters that are capitalized;
14. Number of acronym letters that are capitalized.

❖ C

Appendix: List of acronyms

AC - Acronym candidate
AFP - Acronym Finding Program
ATR - Automatic Term Recognition
CGI - Common Gateway Interface
DB - Database
DC - Definition candidate
DP - Dynamic Programming
HMM - Hidden Markov Model
IE - Information Extraction
LCS - Longest Common Subsequence
LF - long form
MBL - Memory-Based Learning
ME - Maximum Entropy
MeSH - Medical Subject Headings
ML - Machine Learning
NER - Named Entity Recognition
NLP - Natural Language Processing
NPs - Noun Phrases
OCR - Optical Character Recognition
POS - part-of-speech
SaRAD - Simple and Robust Abbreviation Dictionary
SF - short form
TiMBL - Tilburg Memory-Based Learner
TLA - Three Letter Acronyms
TnT - Trigrams'n'Tags
UMLS - Unified Medical Language System