A (ACRONYMS)

by

Manuel Zahariev
Diploma de Inginer, Faculty of Control Systems and Computers,
Polytechnic Institute of Bucharest, 1991

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

In the School
of
Computing Science

Simon Fraser University
April 2004

APPROVAL

Name: Manuel Zahariev

Degree: Doctor of Philosophy

Title of thesis: A (Acronyms)

Examining Committee: Dr. William S. Havens
Chair

Dr. Veronica Dahl, Senior Supervisor

Dr. Fred Popowich, Supervisor

Dr. Maria Teresa Taboada, Supervisor

Dr. Nick Cercone, External Examiner,
Dean, Faculty of Computer Science,
Dalhousie University

Dr. Anoop Sarkar, SFU Examiner,
Assistant Professor, Computing Science,
Simon Fraser University

Date Approved: April 2, 2004

SIMON FRASER UNIVERSITY

Partial Copyright Licence

The author, whose copyright is declared on the title page of this work, has granted to Simon Fraser University the right to lend this thesis, project or extended essay to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users.

The author has further agreed that permission for multiple copying of this work for scholarly purposes may be granted by either the author or the Dean of Graduate Studies.

It is understood that copying or publication of this work for financial gain shall not be allowed without the author's written permission.

The original Partial Copyright Licence attesting to these terms, and signed by this author, may be found in the original bound copy of this work, retained in the Simon Fraser University Archive.

Bennett Library
Simon Fraser University
Burnaby, BC, Canada

ABSTRACT

Acronyms are a significant and the most dynamic area of the lexicon of many languages. Building automated acronym systems poses two problems: acquisition and disambiguation. Acronym acquisition is based on the identification of anaphoric or cataphoric expressions which introduce the meaning of an acronym in text; acronym disambiguation is a word sense disambiguation task, with expansions of an acronym being its possible senses.

It is proposed here that acronyms are universal phenomena, occurring in all languages with a written form, and that their formation is governed by linguistic preferences, based on regularities at the character, phoneme, word and phrase levels.

A universal explanatory theory of acronyms is presented, which rests on a set of testable hypotheses, and is manifested through a set of violable, ordered rules. The theory is developed based on examples from fifteen languages, with six different writing systems. A dynamic programming algorithm is implemented based on the explanatory theory of acronyms. The algorithm is evaluated on lists of acronyms-expansion pairs in Russian Spanish, Danish, German, English, French, Italian, Dutch, Portuguese, Finnish, and Swedish and achieves excellent performance.

A two-pass greedy algorithm for automatic acronym acquisition is designed, which results in good performance for specific domains. A hybrid, machine learning algorithm – using features generated through dynamic programming acronym-expansion matching – is proposed and results in good performance on noisy, parsed, newspaper text.

A machine learning algorithm for acronym sense disambiguation is presented, which is trained and evaluated automatically on information downloaded following search engine lookup. The algorithm achieves good performance on deciding whether an acronym occurs with a certain sense in a given context, and good accuracy when picking the correct sense for an acronym in a given context.

All algorithms presented allow for efficient, readily usable implementations that can be included

as components in larger natural language frameworks. Technologies developed have applicability beyond acronym acquisition and disambiguation, to aspects of the more general problems of anaphora resolution and word sense disambiguation, within information extraction or natural language understanding systems.

to you

Contents

List of Tables

List of Figures

Acknowledgments

I would like to thank my wife Mihaela, for her constant support and encouragement, without which I neither would have contemplated, nor had been able to achieve this. (ti)

I must also thank my parents: Mariana and Alexander Zahariev, who have made me think that pursuing knowledge (and a Ph.D. degree) is a good thing.

I express my deepest gratitude to the following people (I apologize for unwanted omissions):

- Veronica Dahl, my senior supervisor, for her mentorship, patient guidance and exceptional level of academic and moral support.
- Fred Popowich and Maite Taboada (supervisors) for their continuous guidance and support, and for helping me enter and explore the wonderful world of Natural Language Processing.
- My external examiners: Nick Cercone (Dean, Computer Science, Dalhousie University) and Anoop Sarkar (Computing Science, SFU) for their excellent suggestions and guidance.
- Stella Atkins, for her early encouragement (after my presentation of a poster) to pursue acronyms as a thesis topic.
- Bob Hadley, Arvind Gupta and Hassan Aït-Kaci for their guidance in advanced areas of computing science.
- My professors from UPB (Romania): Cristian Giumale, Irina Athanasiu and Dan Suciu, to whom I owe a lot of who I am today.
- My highschool teachers Ştefan Calistrate and Maria Sorescu, who have encouraged disciplined thinking in me and many others.
- Ron Norman and the great team from Inglenet for their encouragement to pursue this degree while working, and for their understanding.

- Nick Miller, Rick Emery and everybody at DataLink Systems for an excellent opportunity to work on really cool products.
- Everybody in the "Logic and Functional Programming Lab" and the "Natural Language Lab" at SFU for their support and for useful, challenging conversations.
- Kersti Jaager and all other staff members at the School of Computing Science, for their help throughout all the years I spent in the program.
- My students from CMPT-307 and CMPT-212, who gave me good reasons not to give up, during the most difficult part of this endeavor.

I would like to thank the following people who have helped me with issues related to acronyms:

- Peter Flynn from the University College of Cork, Ireland for providing the WWWAAS database for the purposes of this research.
- Jens Happe for excellent insights into acronym formation in German.
- Dan Schmelzer and Philippe Westreich for examples and writing of acronyms in Hebrew.
- Maryam Samiei for examples and writing of acronyms in Farsi and for writing Arabic.
- Rocky Yingjian Zhang and Zangjian Hu (Roy) for examples and writing of acronyms in Chinese.
- Satoru Kawai and Mayu Ishida for their patient guidance and help with understanding acronyms in Japanese.

This work was supported in part by:

- SFU Graduate Fellowship Awards.
- NSERC Discovery Grant awarded to Veronica Dahl.
- NSERC Discovery Grant awarded to Fred Popowich.

I would like to thank the open source community, for good quality software tools and hope for the future of software.

I would like to thank "Simon Fraser University": a great school !

Chapter 1

Introduction

Acronyms are universal phenomena of systematic abbreviation of expressions, and represent the most productive source of new lexicon items for many languages. A consistent terminology for describing acronyms and related phenomena has been missing, which has lead to difficulties in the modeling of the human capacity to generate and understand acronyms, and in the design, analysis and evaluation of automated acronym systems.

All prior work has been exclusively focused on acronyms in the English language, even if numerous other languages contain large, dynamic bodies of acronyms. This work is a systematic approach to acronym-related issues, in multiple languages.

The performance of systems presented or discussed is reported using the following measures, considering a term identification task and, respectively, a decision task:

- $\text{precision} = \dfrac{\text{correctly identified terms}}{\text{total number of identified terms}}$
- $\text{recall} = \dfrac{\text{correctly identified terms}}{\text{total number of terms}}$
- $\text{accuracy} = \dfrac{\text{number of correct decisions}}{\text{total number of decisions}}$
- $F_{\beta=1} = \dfrac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$
- $f = \text{precision} \cdot \text{recall}$

1.1 Reviewed acronym systems and methodologies

The work of Cannon, [19] is an extensive overview of acronyms and abbreviations from a historical and linguistic perspective.

Automatic methods for acronym acquisition have been attempted as early as 1995. Taghva and Gilberth [132] propose a method based on inexact pattern matching applied to text surrounding the possible acronym. Their system achieves a good level of success (93% recall and 98% precision) on text from a restricted domain: government studies relevant to the Yucca Mountain Waste Disposal Project. Their methodology is based on the maximization, through a dynamic programming algorithm, of a confidence function related to an inexact match between the first letter in words in expansion candidates and letters in the acronym. I sumbit that this approach, even if yielding high precision/recall for the test corpus, sacrifices potentially important information, contained by letters inside words in the expansion. As a result, some expansions are not counted by the authors, such as DOP for "dioctyphthalate", or are noted as "unusual", such as IGSCC for "Intergranular stress-corrosion cracking".

Yeates [149] uses a heuristic-based system called TLA (Three Letter Acronym), enhanced [150] with an intriguing compression-based method that suggests similarities with statistical collocation discovery approaches. Both systems are evaluated on small, restricted data sets (10, respectively 150 computer science technical reports), the results are not reproducible, the performance is not consistently reported and is relatively low: calculated here as $F_{\beta=1}$=77.83%, respectively 84.71%.

Larkey et al. [82] use combinations between "canonical" and "contextual" approaches in a spectrum of 4 different algorithms. The canonical algorithm attempts to match acronym definitions (acronym-expansion) by looking for definition patterns, such as "Expansion (Acronym)", as in the example: "Department of Defense (DoD)"

The contextual algorithm attempts to match prefixes and up to 4 letters from all words in the expansion with consecutive letters in the acronym. The highest performance, on the $F_{\beta=1}$ measure achieves 92% precision and 88% recall, considering acronyms longer than 2 letters and excluding distances longer than 20 words between an acronym and its expansion.

An excellent evaluation discussion is also included, quantifying the usefulness of automated acronym discovery, by comparing the results from their database with results obtained from hand-crafted acronym databases available online.

Sproat et al. [129] study the problem of unknown (or *non-standard*) word normalization. The term *non-standard word* is used with the same meaning in this thesis, as part of the taxonomy of

acronyms and related concepts, presented in section 2.4. Sproat et al. introduce a series of supervised and unsupervised algorithms based on n-gram models, which predict with high accuracy (91.8–98.1%) tags (associated with tokens in their own taxonomy) of non-standard words, and expand abbreviations (including *some* acronyms) with an accuracy[1] of 93.3–95.2% for supervised methods (where expansions of all abbreviations are manually provided) and 63.5–80.5% for six different, gradually more performant heuristics-based unsupervised methods (expansions of abbreviations are *inferred* from text in the domain). One important shortcoming of Sproat et al.'s method is their treatment of acronyms which are "tokens to be treated as words", such as NATO, which — since they do not need to be *read as their expansions* — are not expanded, and, consequently not entered into the evaluation.

In the medical domain, acronyms (the main representative of the class of *abbreviations*) have received special attention in since, as noted by Pakhomov, [112] (emphasis added):

> ... the Mayo Clinic regards the proliferation of abbreviations and acronyms with multiple meanings as *a serious patient safety concern* and makes efforts to ensure that only the "approved" abbreviations (these tend to have lower ambiguity) are used in clinical practice.

Medical abbreviations (163,666 terms, most of them acronyms) are collected [88] using a heuristic system from the UMLS Metathesaurus [105], an electronic resource of medical terminology in English. The high performance of the system, $F_{\beta=1}$=96.74% is largely a result of significant manual effort that has gone in creating the terminology.

A tool for computer-assisted acquisition of protein name abbreviations [151] has also been proposed, which uses a rule-based recognizer of protein names and a rule-based validation module. The system achieves $F_{\beta=1}$=96.56% reported only on so-called *paranthetical paraphrases*, or short distance acronym definitions where the acronym, in brackets, follows the expansion.

Similarly, the acquisition of acronyms from the molecular biology domain is studied [107], where a rule-based system is used with abstracts of MEDLINE articles [109]. The system achieves $F_{\beta=1}$=82.12%, based on very good figures for precision (93.94–98.76%), but much lower for recall (72.94%, only reported for 55 abstracts of a total 6323 used in the evaluation). The precision of the system increases with the increase in size of the evaluation corpus from 93.93% for 50 abstracts containing 85 introduced acronyms to 98.76% for 6323 abstracts. Since the method uses a secondary phase of *termhood filtering*, which compares the occurrence probability of the term

[1] The error rate reported in the original is converted here to accuracy.

(acronym, expansion or variants) with average values, weighted by the length of the expansion, precision increases with an increase in the size of the corpus, which is expected to involve a dramatic decrease of the recall (not reported) when more than 50 abstracts are used. For 50 abstracts, the number of acronyms correctly found per document is 1.24, decreasing to 0.366 for 6323 documents. I submit that this is also due to use of the same acronym form in multiple abstracts, but raises serious questions about the ability of the method to acquire a significant portion of the acronyms defined in medical abstracts.

The problem of abbreviation normalization (acronym disambiguation) in the medical domain is studied by Pakhomov, [112] using a maximum-entropy (ME) approach, which achieves accuracy slightly below 90% on a corpus of 10,000 reumatology notes. The method uses terms in a two-word window around each occurrence of an acronym as features for the disambiguation, as well as non-linguistic context (section heading on the report form). I suggest that using a larger context window will yield better results.

The automatic acquisition of sense-tagged corpora is also studied, [87] using a method which replaces paranthetical expressions of the type "Acronym (Expansion)" with sense-tagged occurrences of "Acronym" and uses a naïve Bayes classifier for word sense disambiguation. The resulting corpus, with coverage of 92.9% precision and 47.7% recall (96.8% precision and 50.6% recall after removing rare senses and consolidating related senses) can the be used for machine learning of features of abbreviation (acronym) occurrences for a subsequent task of abbreviation normalization.

I finally mention a number of manually built dictionaries of acronyms, including the World Wide Web Acronym and Abbreviation Server[2] and Acronym Finder[3], containing extensive lists of acronyms from many domains. At this moment in time, these sources provide Internet users with the most significant coverage of general purpose acronym searches, with the consolidated lists of links from Opaui Guide to Lists of Acronyms, Abbreviations, and Initialisms[4] providing a good starting point.

1.2 A modular approach to acronyms

Compared to the reviewed methodologies, which only address one problem, usually in one domain and one language (English), this thesis defines and follows a systematic approach to acronym-related

[2]On the web at: http://www.ucc.ie/cgi-bin/acronym (February 2004).

[3]On the web at: http://www.AcronymFinder.com (February 2004).

[4]On the web http://spin.com.mx/~smarin/acro.html (February 2004).

issues in different domains and languages.

The following tasks are identified, also illustrated as modules in Fig.1.1:

(A) Acronym Identification (determine whether a given term is an acronym or not)

(D) Definition Identification (identify possible expansions of acronyms in context)

(M) Acronym – Expansion Matching (match an acronym to a possible expansion)

(O) Acronym Occurrence Disambiguation in Context (disambiguate acronyms in context)

1.3 Thesis format

The thesis is presented in the *article-style* format, in accordance with Simon Fraser University regulations for theses [86, Section 19b] (quote below), [40].

> If several papers are to be presented as a thesis, there must be an abstract, an introduction, and a concluding discussion which ties the work together as a whole.

The article-style format has been chosen in order to present already published results as shorter, stand-alone units, more manageable for the reader. Chapter 3 is included as a stand-alone article, without modifications. Chapters 4 and 5 are included as stand-alone articles, with minor modifications. Each of these chapters contain their own abstract, introduction and references.

Most chapters can be read independently, and follow a common terminology introduced in Section 2.4. The necessary terminology is introduced (and in many cases repeated) wherever used in the stand-alone articles. A common theoretical foundation for the whole work is introduced and argued in Chapter 2.

1.4 Structure of the thesis

A common taxonomy for acronyms and related phenomena, including differentiation tests, is presented in Chapter 2. The tasks of automated acronym acquisition and disambiguation are defined. A universal explanatory theory of acronyms is also presented in Chapter 2, together with supporting examples in fifteen languages using six different writing systems. The theory is supported by a

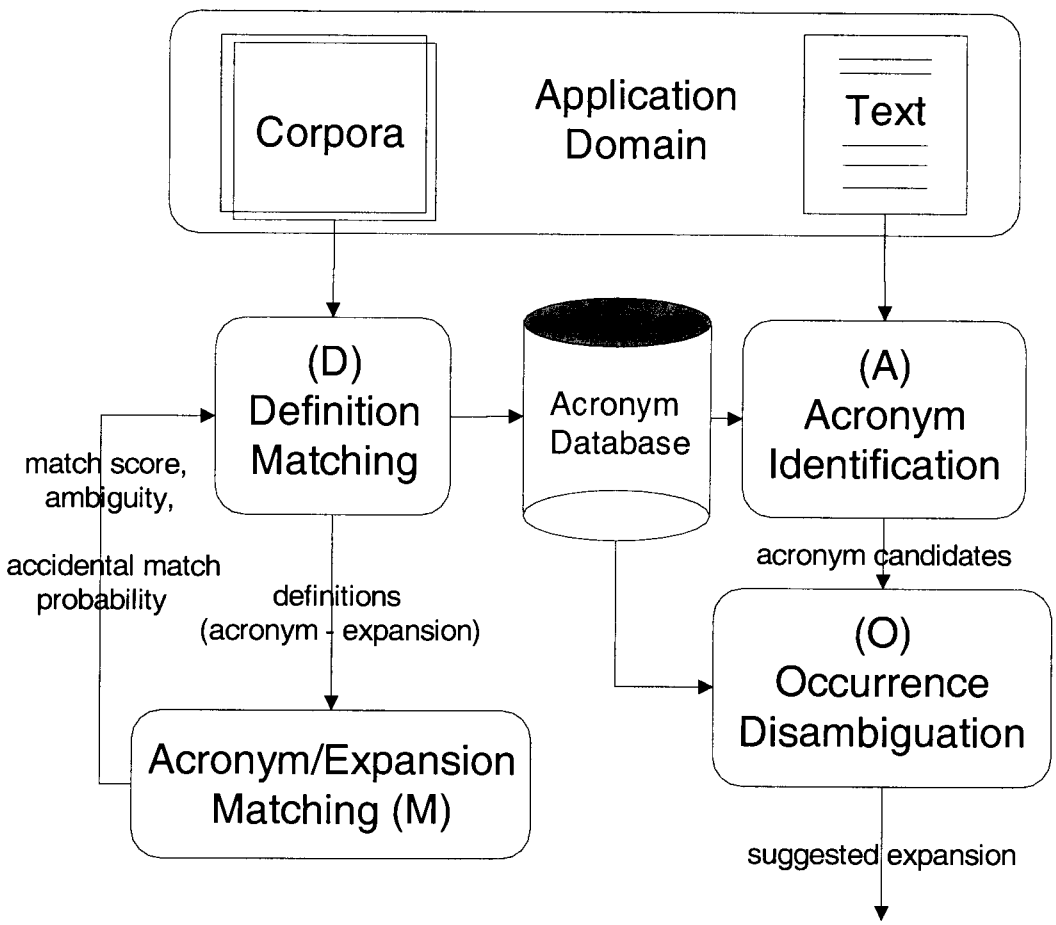


Figure 1.1: A modular approach to acronyms (Figure by author)

number of testable hypotheses, and is formulated in terms of universal hierarchical violable rules, with different orderings and weights in different languages.

A review of Natural Language Processing and Computational Linguistics literature, focused on information useful for automated acronym acquisition and disambiguation, is presented in Appendix A.

An acronym acquisition system using linguistic heuristics in a logic programing formulation of a greedy algorithm is presented in Chapter 3, also published [154]. The system achieves good performance on technical text, and indicates the adequacy of the view of acronyms as linguistic phenomena.

A modular approach to acronym acquisition is proposed in Chapter 4, also published [153] and an acronym-expansion matching component of such a system is built, which uses the universal acronym theory introduced in Chapter 2. Excellent performance of this system measured for acronym-expansion pairs in the English language indicates the adequacy of the theory for acronyms in English. Good efficiency of the algorithm makes it adequate for inclusion as a module in acronym acquisition systems.

A hybrid acronym acquisition system is presented in Chapter 5, also published [152] which uses as a module the dynamic programming acronym-expansion matching system presented in Chapter 4 and automatically learns long-distance acronym definition regularities from feature-value pairs associated with acronym-expansion matchings and from cooccurrence information returned from web search engines. The system achieves good performance on relatively noisy newspaper text in the English language.

A review of difficulties and regularities of acronyms is presented in Chapter 6. Difficulties are found to be characteristic of sparse data problems, common in natural language processing. The fit of the universal acronym theory is evaluated on acronym-expansion pairs in eleven languages. The high level of success observed in the evaluation indicates the adequacy of the theory in modeling acronyms across multiple languages and the direct applicability to other languages of acronym acquisition methods developed for English.

The problem of acronym disambiguation in general text is presented in Chapter 7, as a specialized instance of the word sense disambiguation task. An unsupervised machine learning solution to this problem is described, which starts from acronym dictionary entries and is automatically trained on examples downloaded from results of search engine queries. The system is evaluated on a number of acronym forms with multiple senses and its performance is found to be good and consistent across different acronym forms.

Chapter 8 briefly presents conclusions and future work.

1.5 Summary of contributions

The thesis proposes in Chapter 2 a novel, common taxonomy of acronyms, their expansions and related phenomena, such as non-standard words [129], abbreviations, direct abbreviations, creative spellings [102], names and code names. A number of tests are proposed in section 2.4, that can differentiate between the ambiguous categories of *acronyms* and *direct abbreviations*, and respectively between *acronyms* and *code names*.

The main value of the taxonomy is in defining in a problem-independent way the nature and semantic boundaries of the term acronym, which can lead to a modularized architecture and evaluation of systems dealing with acronyms and, respectively related terms.

A universal theory of acronym formation is proposed in section 2.5, which is based on seven hypotheses:

- Acronyms are *universal* phenomena (hypothesis 2.7).
- *Short acronyms* are less prefered due to the incidence of accidental matches (hypothesis 2.1).
- Acronyms with higher *pronunciability* are prefered (hypothesis 2.2).
- *Conflict* between multiple ambiguous acronyms is avoided (hypothesis 2.3).
- *Syntactic acronym inversion* occurs with expansions with overlapping syntactic constituents (hypothesis 2.4).
- *Morphological acronym inversion* occurs with different intra-word morpheme and word order (hypothesis 2.5)
- *Accidental inversion* occurs with alternate semantically equivalent expansions (hypothesis 2.6).

A set of fifteen rules for acronym formation are proposed, based on the hypotheses:

- Five letter matching rules (initial: rule 2.1, morpheme: rule 2.2, syllable: rule 2.3, group: rule 2.4, bare letter: rule 2.5).
- Three word skipping rules (link word: rule 2.6, punctuation: rule 2.7, regular word: rule 2.8).

- Three lexical rules (inflection: rule 2.12, migration: rule 2.11, import: rule 2.15).
- Two *special* rules (plural duplication: rule 2.9, symbolic matching: rule 2.10).
- Two directional rules (consecutive matching: rule 2.13, inversion: rule 2.14).

Acronym formation rules are considered to be universal, but their ordering (weights) is different from language to language, leading to a computationally efficient implementation (using dynamic programming) of an acronym-expansion matching algorithm presented in Chapter 4. The algorithm aligns strings represented by an acronym and its expansion using weigths associated with applications of each acronym formation rule. Using this algorithm, chapters 4 and 6 propose orderings (and weights) of the acronym formation rules for English (evaluated on an acronym dictionary) and ten other languages (evaluated on multilingual acronym lists and a Russian abbreviation dictionary), which achieve very high performance ($F_{\beta=1}$ 98.58% for Russian and over 99% for for Spanish, Danish, German, English, French, Italian, Dutch, Portuguese, Finnish, and Swedish) when matching acronyms with their expansions.

The difficulties of acronym problems are quantified in chapter 6 by a set of metrics: matching ambiguity, accidental matching probability and polysemy. The distributions of acronym matching ambiguity and accidental matching probability on a set of 894 acronym definitions in English (randomly extracted from an acronym dictionary) are found to be consistent with Zipf-Mandelbrot distributions found in other natural language problems, indicating similarities with sparse data problems. The polysemy of acronyms in an acronym dictionary with 17,529 entries is found (in section 7.2) to be similar and more productive than that of regular words from WordNet (hypothesis 7.1). Following the success of the acronym–expansion matching algorithm based on alignment presented and evaluated for English in chapter 4 and evaluated for English and ten other languages in chapter 6, the acronym formation problem is found to have optimal substructure, governed by different orderings of the rules presented in section 2.5 for different languages. The optimal substructure has computational implications leading to efficient implementations of acronym–expansion matching systems compared to the exponential complexity of a brute-force approach.

Using a subset of the acronym formation rules, Chapter 3 presents a heuristic, monolithic two-pass approach to acronym acquisition, which achieves good performance ($F_{\beta=1}$ 95.22% on 1470 documents) on a corpus of abstracts of technical documents. The main contribution of this chapter lies in the second pass, which attempts to find acronym expansions in a larger window around the acronym occurrence, improving recall, without a sacrifice in precision. It is proposed that this

method can improve the performance of other existing monolithic acronym acquisition systems when used in combination with them.

A modularized approach to acronyms is proposed in section 1.1, with three non-trivial modules:

- Acronym–expansion matching.
- Acronym definition matching.
- Acronym disambiguation.

Acronym–expansion matching is achieved by the component presented in Chapter 4. Given its high performance (precision and recall), its use as a module in an acronym acquisition system is apropriate. The *alignment parameters* returned by this module (as a result of application of the acronym formation rules on acronym–expansion candidate pairs) are used as features that a machine learning acronym definition matching module can be trained on. Such a modular solution (acronym–expansion matching through alignment and machine learning of acronym definition matching) is proposed in chapter 5 for acronym acquisition, viewed as a subset of the anaphora/cataphora resolution problem. Acronym definitions are anaphoric/cataphoric expressions, and matching restrictions (such as gender and number in the case of pronoun anaphora) are provided by the parameters of the alignment of acronym and expansion candidates. It is shown that the regularities of such definitions (even long-distance) can be learned from manually marked corpora, achieving much better performance ($F_{\beta=1}$=92.38% on 400 documents representing news stories) than pronominal anaphora resolution systems on newspaper text.

Acronym disambiguation is viewed as a restriction of the word sense disambiguation problem in chapter 7, and an unsupervised machine learning methodology is proposed for it. The resulting system learns from terms occurring in the same document as the acronym, using smoothed distance-based word occurrence features. Fully automatic acquisition of training examples is achieved using documents downloaded following the results of search engine queries, starting from a dictionary of acronyms. Fully automated evaluation of the system is achieved by substituting occurrences of expansions of sought acronyms with the acronyms (sense-marked with the expansion). The method is based on the hypothesis of "one sense per document", widely used in word sense disambiguation (hypothesis 7.3). The resulting system achieves accuracy over 92.58% at picking the correct sense for an acronym in a given document and $F_{\beta=1}$=91.52% at deciding whether a given sense of an acronym matches a given occurrence. Further, a method that can automatically identify close or interacting senses of acronyms is presented, which improves the performance even further.

A discussion, supported by examples, of acronym formation regularities in fourteen languages other than English (Spanish, French, German, Finnish, Italian, Hungarian, Romanian, Russian, Bulgarian, Hebrew, Arabic, Farsi, Chinese and Japanese) with six different writing systems is provided in chapter 2.6. A quantitative evaluation of the fit of the universal theory of acronyms (acronym formation rules) introduced in section 2.5 is provided in section 6.5 on a set of eleven languages (Russian, Spanish, Danish, German, English, French, Italian, Dutch, Portuguese, Finnish, and Swedish). The theory is found to account for all acronyms in an evaluation set based on a multilingual list of acronyms, and on random entries from a Russian abbreviation dictionary. The computational solutions for acronym acquisition presented in chapters 3, 4 and 5 are, consequently, directly applicable to all these languages. Lexical matching rules (migration, inflection), which are recognized but are not modeled linguistically in this thesis, are only necessary for a small subset of acronyms in a small subset of languages.

Chapter 2

A Taxonomy for Acronyms and Related Phenomena

This chapter presents a consistent terminology for acronyms, abbreviations and related terms, to help in the construction, analysis and evaluation of systems that use acronyms and abbreviations, with coverage in multiple languages.

Acronym formation regularities are consolidated into a universal explanatory theory of acronyms, based on a number of hypotheses and presented as a universal set of violable rules. Different orderings and the enabling or disabling of certain rules in the set are applicable to specific languages.

Acronym formation regularities are discussed and examples are provided for a number of languages: English, Spanish, French, German, Finnish, Italian, Hungarian, Romanian, Russian, Bulgarian, Hebrew, Arabic, Farsi, Chinese and Japanese.

2.1 Motivation

Acronyms represent significant barriers for humans to understanding specialized text, as well as for the automatic processing of natural language.

The size of the problem is significant. The most comprehensive hardcopy dictionary of acronyms for the English language [13] lists 450,000 entries, most of which are acronyms. This publication is currently at its thirty-second edition, with four editions in only the last three years. Acronym Finder[1], an online resource, claims coverage of over 330,000 acronyms, with an average of 196

[1] http://acronymfinder.com (February 2004).

entries being added every day. Specialized acronym dictionaries are also available, as are numerous domain-specific online lists of acronyms.

New acronyms, such as 'SARS' for "Severe Atypical Respiratory Syndrome", are constantly being created, which justifies a need for software tools that can assist with the automatic acquisition of acronyms from large text collections.

Acronyms occur in numerous languages, with diverse writing systems. Most existing acronym resources have a predominant English language focus. Given the high cost of systematic manual acronym acquisition, this creates a need for accurate and efficient multilingual automatic acronym acquisition systems.

Acronyms are highly polysemous. At the time of this writing Acronym Finder listed 64 different senses for 'CIA'. Such occurrences can result in significant difficulties for humans when trying to identify the use of less popular senses. Acronym polysemy creates a need to assist users with the automatic disambiguation of the meaning of acronyms in the context of their occurrence.

Beyond acquisition and disambiguation, a good understanding of phenomena related to the formation and usage of acronyms is a prerequisite to accurate information extraction from documents containing acronyms and also necessary for automated understanding.

2.2 Historical Perspective

The word "acronym" has Ancient Greek etymology: akron (=end, also tip), onoma (=name) [131], in loose interpretation "something formed from word ends". The term acronym is credited [19, page 107] to an unnamed Bell Laboratories researcher.

The creation of acronyms is a predominantly written language phenomenon, being justified by better use of the real estate of a document, and by saving writing and typing time for the repetitive use of a long expression. In ancient times, when words were carved in stone, that amounted to significant savings in the effort of scribes/sculptors.

Acronyms have a two-millenary history. The Republic of Rome commonly used 'SPQR', an acronym for "Senatus Populusque Romæ" with meaning "the Senate and the People of Rome"; note the conjunction 'que', included in a a morphological compound, illustrating the idea that acronym letters commonly match intra-word morpheme (not just word) initials.

Another acronym is shown in various early Christian depictions of the Crucifix: 'INRI', an acronym for "Iesus Nazarenus Rex Iudæorum" (as explained in *The Gospel of John* 19:19-20 in the New Testament), meaning "Jesus of Nazareth, King of the Jews". Similar depictions in the Greek

language use 'INBI' for "Iésous o Nazóraios o Basileus tón Ioudaión".

A number of other Latin acronyms are first attested in European Medieval and Renaissance writings, such as 'QED' for "Quod Erat Demonstrandum", meaning "which was to be demonstrated". The origin of this acronym is credited to the rationalist philosopher Benedictus de Spinoza (1632-1677), in his *Ethica More Geometrico Demonstrata.*

In spite of their two-millenar history, acronyms see significant proliferation only in the twentieth century.

Acronyms are an important source of new language elements. They go through an evolutionary process starting with the abbreviation of commonly-used multi-word constructions, such as 'BSE' for "Bovine Spongiform Encephalopathy". With time and extensive use, acronyms become standalone concepts, such as 'VCR', from "Video Cassette Recorder", arguably used often without reference to its component words (e.g. recorder).

Sometimes, the necessary implication of the original compositional elements in an acronym is lost altogether, as is the capitalization, as in the case of 'radar', an acronym for "Radio Detection And Ranging". Most teenagers associate 'laser' with a "desirable light-emitting futuristic weapon" rather than consider it an acronym for "Light Amplification by the Stimulated Emission of Radiation".

Finally, acronyms migrate spelling into phonetically similar forms and lose their relationship with the original expression completely. They become — truly — new words. The word 'okay', migrated phonetically from 'OK', which originates from an electoral slogan, an abbreviation for "Old Kinderhook", NY, the birthplace of US president Van Buren [131].

2.3 Terminological Perspective

One of the important difficulties in the automatic processing of acronyms is the lack of a consistent terminology and of universally accepted evaluation methodologies.

Most commonly used definitions of acronym as "word formed from the initials of other words," [131] fail to account for many acronyms, such as 'DNA' (for "deoxyribonucleic acid"), 'XML' (for "Extended Mark-up Language"), or even the ancient SPQR.

Some acronym acquisition systems such as Taghva and Gilbreth [132] avoid difficult expansions such as 'DOP' (for "dioctyphthalate"), by using a narrow definition of acronym, which results in increased reported performance.

In the evaluation of their system Larkey and colleagues [82] only account for acronyms of three letters or longer, and for situations where the acronym is closer than twenty words away from the

expansion, resulting, again, in increased reported performance.

Task-specific systems approach the problem of *text normalization*, i.e. substitution of so-called *non-standard word* with their normalized correspondents for text-to-speech applications [129] or the substitution of abbreviations with their expansions, for information retrieval [137], inter alia. Such systems include acronyms, either openly or implicitly, in wider categories, without an explicit definition of their nature or a systematic exploration of their properties.

Further complications in the interpretation of acronyms arise from informal language phenomena such as *creative spellings* [103] which occur frequently in some domains, such as online chat or short wireless messaging.

A comprehensive review of literature (also historical texts) on and containing abbreviations, including acronyms is provided by Cannon [19]. A taxonomy of such phenomena is also presented ibid., with a linguistic focus on the formation and usage of abbreviations (called *shortenings* by the author). I submit that this taxonomy is not readily usable in computational solutions to acronyms and abbreviations, since the boundaries of the concepts are not delimited enough to allow use in automated systems, and are often left to individual interpretation. Quantitative distinctions such as indicated below are useful from a linguistic point of view, but I submit they are too fine-grained to serve well computational solutions.

> Not more than two initial letters/sounds of some or all of the constituents can be retained, though an exception of three, or even four is permitted if the majority of the reduction typifies acronymy.

A comprehensive definition of acronyms, applicable across multiple languages and domains, as well as the delineation of clear boundaries with related concepts will help in the construction and evaluation of future acronym systems.

A consistent terminology for acronyms and related phenomena is proposed further. Differentiation tests are suggested, with the goal to help isolate various aspects of the automatic processing of acronyms, abbreviations and related terms. In this account, the treatment is restricted to correctly spelled or written terms, but mis-spelling is recognized as an additional challenge to be overcome by any practical systems.

2.4 Terminology

This section introduces a consistent set of definitions covering acronyms and related concepts, in order to solve one of the most important difficulties of acronym systems: the lack of a consistent terminology and blurry delimitations between terms.

Definition 2.1 (Non-standard words) Non-standard words *are terms that do not occur* ad literam *as entries in dictionaries, but have semantic value that can be reliably recognized by humans from specific socio-linguistic groups.*

Non-standard words are defined functionally by Sproat et al. [129] as terms not found in dictionaries, for which text-to speech systems cannot be trained. A taxonomy for such terms is proposed ibid., which fits closely *solutions* presented by the authors to non-standard word normalization problems arising in the text-to-speech domain.

Acronyms fall into four separate categories within the [129] taxonomy, as well as outside of it:

- Abbreviations (coded EXPN). The example 'N.Y.', arguably an acronym (although read as "New York") is said to fall into this category.
- Letter sequences (coded LSEQ). The acronyms 'CIA', 'D.C.' (for "District of Columbia") are included as examples in this category.
- Read as word (ASWD). I submit that terms as 'SQUID', an acronym for "Superconducting Quantum Interference Device", which is also read as the English word 'squid' (**skwɪd**) fall into this category.
- Mixed or split (SPLT). Acronyms such as '3M' for "Minnesota Mining and Manufacturing Company" fall into this category.
- Outside of the Sproat et al. taxonomy. For example, WordNet [37] lists as an entry the acronym 'WHO' for the "World Health Organization" which is not — consequently — a non-standard word for applications using WordNet.

The taxonomy of Sproat et al. is mixed, since it contains items grouped by *what they are* (resulting in how they should be pronounced), such as addresses (NADDR), telephone numbers (NTEL), zip code or PO Box (NZIP), together with items grouped by *how they should be pronounced*, such as ASWD, LSEQ above.

A taxonomy for acronyms and related concepts is proposed here, based on *what they are*, as opposed to *how to deal with them* which results in clearer classifications for *how to handle them* within systems that address specific problems, such as information retrieval, text-to-speech, information extraction, etc.

Figure 2.1 illustrates, as a Venn diagram, the proposed taxonomy for abbreviations, acronyms and related terms.

Definitions follow below. Non-standard words are represented as the complement of the set of "Dictionary Coverage", which contains all words covered in dictionaries. The set "Dictionary Coverage" is dynamic, since more acronyms and abbreviations are constantly being added to dictionaries, as they penetrate the mainstream of language. The set of "abbreviations" has three subsets: symbolic abbreviations (no relationship to the expansion), direct abbreviations (ad-hoc, easily recognizable) and acronyms (systematic abbreviations, the object of this thesis).

Initialisms and syllabisms are special cases of acronyms. Non-standard words overlap with all presented categories of abbreviations. Creative spellings overlap with symbolic abbreviations and acronyms (when using symbolic matching, rule 2.10 below). Code names overlap with acronyms.

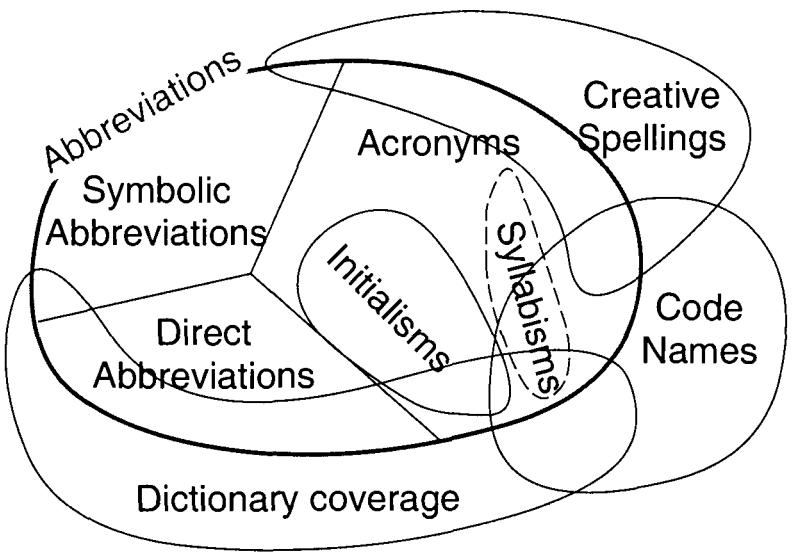


Figure 2.1: Acronyms, abbreviations and related terms (Figure by author)

Definition 2.2 (Abbreviation, expansion) Abbreviations *represent substitutions of shorter terms for words or expressions (called* expansions*) through the association of letters, letter sequences, symbols or digits in the abbreviation with words in the expansion.*

Three distinct categories of abbreviations are introduced below, which cover completely the scope of the term "abbreviation".

Definition 2.3 (Direct abbreviation) Direct abbreviations *are simple abbreviations of words or collocations. Depending on their usage domain, direct abbreviations follow strict abbreviation rules, or are invented or used ad-hoc.*

Examples of direct abbreviations are 'abbr.' for "abbreviation", 'drt.' for "direct", or 'drt. abbr.' for "direct abbreviation".

Definition 2.4 (Acronym) Acronyms *are a systematic form of abbreviation for individual words or (more frequently) for complex expressions. In languages which support capitalization, acronyms are generally written using special capitalization patterns which must be non-overlapping with the capitalization of common words and proper names in that language.*

The study of acronyms, as defined above, is the subject and the focus of this thesis.

The capitalization of acronyms as common words usually indicates their acceptance as standalone words into the mainstream of language, at least within the scope of the discourse containing their usage. For example, uses of 'laser', 'radar' and 'ok' need not be capitalized, even if they are acronyms (at least at origin). No capitalization in the use of the acronym 'tanstaafl' (for "There Ain't No Such Thing As A Free Lunch") in Robert Heinlein's science fiction novel "The Moon is a Harsh Mistress" indicates its common use within the described hypothetical society.

The capitalization of acronyms as proper names indicates their use as *names*, as specified in definition 2.8.

The systematic nature of acronyms manifests itself through the following phenomena:

- The repeated use of the same acronym within the same document or across multiple documents within the same domain.
- The repeated applicability of the same matching rule for the same acronym, such as for successive matches of initials of words in the expansion.

- The applicability of the same matching rule for multiple acronyms, such as the substitution of a group of identical matching letters with a digit indicating the cardinality and the letter, as specified by rule 2.10 below.
- The applicability of multiple matching rules for the same acronym.

The following non-strict tests are proposed for differentiating between acronyms and direct abbreviations:

1. When text containing abbreviations is read, direct abbreviations are usually read as their expansions, whereas acronyms are read directly.
2. The expansion of direct abbreviations is generally self evident from context, and does not need to be provided explicitly, whereas acronyms generally need to be defined explicitly, whether in the same text as used or not.
3. Special capitalization (all capital or mixed case) is generally used for acronyms, and not used for abbreviations, which sometimes follow the capitalization patterns that the expansion would have followed, in the particular context of the abbreviation occurrence.
4. Periods or other punctuation is generally necessary within or at the end of direct abbreviations. An acronym retains punctuation (and its sequence) found in the expansion. Periods can be present within the acronym but are generally unnecessary.
5. Explicit capitalization or emphasis patterns can be used with expansions of acronyms, stressing the relationship between letters in the acronym and words in its expansion. Such patterns are not applicable to direct abbreviations.

Definition 2.5 (Acronym definition) *An* acronym definition *is the occurrence in text of an acronym — expansion pair, with the purpose to introduce to the reader the meaning of the acronym, as being the expansion.*

Chapter 5 differentiates between *short-distance* acronym definitions, which can be modeled by a limited number of regular expressions, and *long-distance* acronym definitions, which are a form of discourse anaphora. Short-distance and long-distance acronym definitions obey different regularities, leading to the applicability of different algorithms to matching acronym — expansion pairs within a local context, or within the more general context of a paragraph or a whole document.

Definition 2.6 (Symbolic abbreviations) Symbolic abbreviation *are ad-hoc abbreviations, usually originating in the scientific and technical literature, which have no relation to the expansion, other than one of association.*

For example, 'c' for "speed of light" is a symbolic abbreviation, common in the domain of physics, but also recognizable by members of the general population.

In text following the paragraph in the example below, 'x' is a placeholder for the phrase "the mass of a given object of volume $1m^3$".

> Consider a number of objects of varying volumes and masses. Let x be the mass of a given object of volume $1m^3$.

Symbolic abbreviations generally require special or conventional emphasis patterns within text. For instance, a book on software algorithms can systematically use **bold** for the names of variables.

Definition 2.7 (Initialism) Initialisms *are acronyms pronounced through the consecutive verbalization of their letters.*

'RFC' (for "Request for Comments") pronounced **ɑrefsi**, or 'CLR' (for "Calcium Lime Rust") pronounced **sielɑr** are examples of initialisms. 'SCSI', an acronym for "Small Computer Systems Interface", pronounced **skʌzɪ** is an acronym, but not an initialism.

Definition 2.8 (Name) Names *represent authoritative designations, which have no direct or overt relationship to the nature of the designated item.*

The study of characteristics of names has been the focus of significant debate in the philosophy of language, for example about the relationship between names and descriptions, [124]. A lot of the complexity of the results is avoided here by focusing on the relationship between names, abbreviations and acronyms, and on issues of ambiguity between proper names (definition 2.9), code names (definition 2.10) and acronyms.

An important characteristic of names is the existence of a *naming authority*, with the right to assign or negotiate an undisputed name to an entity through an *act of naming*, similar in substance with the *speech act of naming* described in [7], possibly with requirements of uniqueness within a domain.

As a side effect, names can be found in systematic nomenclatures, managed by naming authorities which record *traces of naming acts*. For example, the names of all Honda motorcycle models are

found in product lists created and managed by their manufacturer — Honda. Similarly, the names (*proper names*, as defined below) of all humans born in a certain region can be found in the records of the authorities which issue birth certificates in that region.

In what follows, *names* are differentiated into two categories: *proper names* and *code names*, the latter occasionally ambiguous with acronyms.

Definition 2.9 (Proper name) Proper names *are* names *of people, geographic places, nameable objects, things, entities, animals and plants, which can be designated through the exclusive use of alphabetic symbols.*

Depending on the conventions of various languages, proper names require the use special capitalization (capital initial) or special emphasis.

Definition 2.10 (Code name) Code names *are* names *which either:*

- *Contain numeric or special character symbols, or*
- *Require different capitalization than that of proper names.*

Code names fall into systematic or standardized nomenclatures, such as ISO, the International Standards Organization[2] country codes, where a requirement for non-collision between the names of all describable unique entities is essential, as is the uniformity (length) of the codes.

Alternately, code names can be trademarked or copyrighted terms or expressions. For example, 'CBR954RR'™ is the code name of a certain Honda© motorcycle model.

Some code names can be acronyms (or can *originate from* acronyms), such as 'NT'™, a code name for a Microsoft© product, and also an acronym for [Windows] "New Technology™."

The following non-strict tests are proposed to distinguish between acronyms and code names:

1. Acronyms are abbreviations of expressions. Only certain code names are abbreviations, and they are also acronyms.
2. Native speakers generally agree as to the match between an acronym and its expansion. Matching with (or the existence of) an expansion is not necessary for code names.

[2]On the web at: http://www.iso.org (February 2004).

3. Code names are a *necessary* way of naming an entity or a class, whereas acronyms are *alternate* ways of naming entities that can also be described by their expansions (which can be names).

4. Code names can be chosen to represent or include acronyms, but their meaning does not rely on, and only marginally represents (if at all) the meaning of the acronym. For example the "new technology" in Microsoft's Windows 'NT' has been since replaced by two successive generations of newer operating systems, without a need to change the model name.

5. Understanding by humans of the origin of a code name is generally considered secondary to the recognition of the entity or class designated by the name. The expansion of an acronym is *the* essential element of the meaning of the acronym.

Definition 2.11 (Acronym form) *An* acronym form *represents a spelling of an acronym. Multiple acronyms (with different corresponding expansions) can share the same acronym form.*

For example, 'CPA' is an acronym form for both the "Canadian Paraplegic Association" and for "Certified Public Accountant".

Definition 2.12 (Creative spelling) Creative spellings *are alternate, many times ad-hoc, ways of spelling existing words or expressions, usually based on similarity of pronunciation, sometimes based on letter or word play and non-textual considerations.*

Creative spellings [103] occur frequently in some domains, such as online chat or short wireless messaging. Examples such as 'UR-L8' for "you are late", the French 'j'm' for "j'aime", and many others can be found in newsgroup messages or in the copy of advertisements.

The name of the latest Microsoft family of operating systems, Windows XP™ , is said to originate from the pre-release name of the internal development project that had it as an end result: *Cairo*. The name was inspired by the similarity in shape of the group of Latin letters 'XP', with the creative spelling of "Cairo" using the English pronounciation of the Greek letters *chi* and *rho* ($\chi\rho$).

Definition 2.13 (Function word) Function words *are prepositions, conjunctions, articles, and particles.*

Evidence is presented further that function words have preferential skipping within expansions of acronyms.

Definition 2.14 (Acronym acquisition) Acronym acquisition *is a lexicon-building process by which acronyms are collected either manually or automatically from textual evidence.*

Definition 2.15 (Acronym sense disambiguation) Acronym sense disambiguation *is a mechanism by which the appropriate sense (expansion) for a specific occurrence of an acronym form in a given context is selected.*

An overview of technologies and systems dealing with acronym acquisition, disambiguation and related problems is presented in section 1.1.

2.5 A Universal Theory for Acronym Formation

One of the main difficulties in designing and evaluating acronym systems is the wide coverage of phenomena which are generally considered to be covered by *acronym formation.* For example, acronym letters do not match only initials of words in the expansion. Many more uncommon scenarios (described further) in many languages are readily qualified by native speakers as *acronyms*, even when the matching is not directional, or does not even involve letter similarities with the expansion.

An exhaustive description of phenomena which lead to acronym formation is proposed in this section, as a set of acronym formation rules. This set of rules is said to be universal, since it accounts consistently for all acronyms known to the author in fifteen languages (English, Spanish, French, German, Finnish, Italian, Hungarian, Romanian, Russian, Bulgarian, Hebrew, Arabic, Farsi, Chinese and Japanese) with representative examples discussed in section 2.6 and for all acronyms in databases and lists in eleven languages (Russian Spanish, Danish, German, English, French, Italian, Dutch, Portuguese, Finnish, and Swedish) for which these rules are automatically found to provide complete coverage (chapters 4 and 6).

I propose here a collection of acronym formation rules, which is an extended version of the one presented in chapter 3. The rules apply on the written form of acronym-expansion pairs and, combined in individual situations, attempt to explain the formation of all acronyms, in all languages. The set contains redundant, and also contradictory rules. Preference for redundant rules is given based on their *ordering* and *weight.* In exceptional situations, contradictory rules mutually disable each other, such as the case of the contradictory rules consecutive match (rule 2.13) and inversion (rule 2.14).

A set of proposed rules is considered to be *universal*, although for every individual language:

- The ordering (importance) of rules is different from the ordering for other languages.

- Some rules are disabled (or not applicable). Rule 2.3, for instance, is not applicable in languages (such as Chinese) where characters encode whole syllables.

Rule 2.1 (Initial matching) *Matching of the initial character of a word in the expansion with an identical character in the acronym.*

The exclusive, repeated application of rule 2.1 accounts for most acronyms, in most languages and encompasses commonly accepted dictionary definitions of acronyms, such as [131].

Hypothesis 2.1 (Short accidental matches) *Longer acronyms matching initials and internal characters in the expansion are preferred over two-letter acronyms matching only initials.*

When abbreviating two-word expressions, the resulting two-letter acronym usually yields a high incidence of spurious matches, leading to increased difficulty in recognizing correctly the expansion in text. As a result, there is a tendency to also match characters within words in the expansion. To illustrate, in example (2.1), 'MTB' is the common (and consequently preferred) acronym for a two-word expression ("mountain bike"), rather than 'MB', a possible acronym for the same expression, obtained from matching only initials.

Hypothesis 2.2 (Pronounciability) *Acronyms with improved* pronounciability as words *(sometimes matching existing words in the target language) by adding matches at internal characters are often preferred over acronyms matching only initials.*

Examples such as (2.5) are common, where 'AVSCOM', matching initials and internal characters, is used (and consequently preferred) over 'ASC', a possible alternate acronym for the same expansion ("Aviation System Command") matching only initials.

I submit here that hypotheses 2.1, 2.2 and 2.3 are the main reasons for matching letters other than initials of words in the expansions of acronyms.

Hypothesis 2.3 (Conflict) *Possible new acronyms which would be identical to acronym forms already used in a given domain are avoided within that domain, leading to matching strategies different from initial matching.*

Rule 2.2 (Morpheme matching) *Matching of the initial character of intra-word morphemes within the expansion with an identical character in the acronym.*

Rule 2.2 is especially productive in languages and domains where morphological compound words are common, such as German, Swedish, Dutch, Russian, Finnish and, respectively, domains such as medical or chemical terminology.

Rule 2.3 (Syllable matching) *Matching of the initial character of an intra-word syllable within the expansion with an identical character in the acronym.*

Rule 2.3 is used for acronym acquisition systems by [88] and here in chapter 3. Examples such as (2.1) and (2.2) support it, but, due to its low incidence (sparse data problem), clear evidence in its favor has not been provided so far.

'MTB' for "Mountain Bike" (2.1)

'MKTG' for "Marketing" (2.2)

Rule 2.3 is *useful*, even if there has been *no evidence* presented so far that people prefer matches on syllable boundaries over than on any other internal letters (it may not even be true that syllable matching is preferred over matching on other letters), as follows. In languages or domains where morpheme matching is common, and for applications lacking access to morphological resources to help with splitting words into morphemes, syllabification provides an usable, although imperfect alternative, since syllable boundaries always follow, but can exceed significantly the number of morpheme boundaries.

For example, considering square brackets '[ ]' for morphemes and round brackets '( )' for syllables, the word 'hypertext' is split: **[(hy)(per)][(text)]**. Lacking morphological resources, syllable matching *approximates* morpheme matching in example 2.3.

'HTTP' for "Hypertext Transfer Protocol" (2.3)

However, syllabification must follow morpheme boundaries, and syllabification systems which do not use morphological resources will fail for words such as **[(po)(ly)][(chlo)(ri)(na)(ted)]**, abbreviated in example (2.4), and incorrectly syllabified: **(po)(lych)(lo)(ri)(na)(ted)**, using English language syllabification rules.

'PCB' for "polychlorinated biphenyl" (2.4)

Rule 2.4 (Group matching) *Matching of a group of consecutive characters in a word in the expansion with an identical group of consecutive characters in the acronym.*

Rule 2.4 can be combined with rules 2.1, 2.2 and 2.3, to account for groups of matching characters at the beginning of words, morphemes and respectively syllables.

'AVSCOM' for "aviation systems command" (2.5)

Group matching is a common occurrence in situations where whole syllables of words in the expansion can be found in the acronym, as in example (2.5), and is very productive in Russian, especially for Soviet terminology, as illustrated in examples (2.55), (2.56) and (2.57), through the formation of *syllabisms*.

Rule 2.5 (Internal character matching) *Matching of an internal character in a word in the expansion with an identical character in the acronym.*

Rule 2.5 is the common denominator of all letter matching rules, and applies in situations where none of the rules 2.1, 2.2, 2.3 or 2.4 are applicable.

Definition 2.16 (Syllabism) *A* syllabism *is an acronym formed exclusively from the repeated application of group matching (rule 2.4) to complete syllables of words in the expansion matching syllables in the acronym.*

Creating acronyms for longer expressions through successive matches of initials or other letters of *all words* within the expansion leads to excessively long acronyms. This tendency is counterbalanced through the *skipping* of specific words in the expansion (not matching letters in the acronym to those words).

Different preference is given to skipping different words, as illustrated below.

Rule 2.6 (Function word skipping) *Skipping altogether a function word in the expansion, when matching with the acronym.*

Function words show the highest skipping preference in all languages and domains, as they do not communicate new meaning contributed to the expansion, but rather tie together the semantic elements of the expansion.

Rule 2.7 (Skipping of a word preceded by punctuation) *Skipping altogether in the expansion a word preceded by certain punctuation (/ –), when matching with the acronym.*

In some languages, such as English and French, words preceded by '/' (slash) or '-' (dash or hyphen) have a higher skipping preference. This phenomenon is less common in Swedish, German or Dutch, which have punctuation rules different than English, and where '-' frequently precedes conjunctions, in constructions that combine an additional morpheme (before the dash) with a morphological compound, where the second morpheme in the compound is implied to form another morphological compound with the morpheme before the dash. In example 2.6 below, "Land- und Forstwirtschaft" is an alternate way of writing "Landwirtschaft und Forstwirtschaft".

'BmLF' for "Bundesministerium für Land- und Forstwirtschaft" (2.6)

Rule 2.8 (Word skipping) *Skipping altogether a word in the expansion, when matching with the acronym.*

Rule 2.8 is the common denominator of all word skipping rules, and its application is common in the abbreviation of longer expressions, or when increased pronounciability is sought for the resulting acronym.

Rule 2.9 (Plural duplication) *Duplication in the acronym of a letter in the expansion, when matching a plural form of a noun phrase.*

Plural duplication is only present in some languages, such as French and Spanish, illustrated in examples (2.25) and, respectively, (2.16) below, and domains, such as medical or scientific text, for Latin-style abbreviation, e.g. 'aa' for "arteries", or 'pp' for "pages".

Rule 2.10 (Symbolic matching) *Matching of a symbol, character, morpheme, group of characters, word or expression in the expansion with a character or group of characters in the acronym, following ad-hoc rules, which are usually recognizable by members of a professional, social or historic environment.*

Symbolic matching can sometimes be represented by the application of *creative spelling* mechanisms during acronym formation, as matching 'U' to "you" and 'C' to "see" in (2.7).

'CU' for "See you" (2.7)

Matching the X to the morpheme 'trans' in (2.8) and the match of 'X' to 'Christ' in (2.9) are other examples of symbolic matching.

'IXFR' for "Incremental zone transfer" (2.8)

'XMAS' for "ChristMAS" (2.9)

Numeric multiplicative matching is a special kind of symbolic matching, where a numeral indicates the *number of matches of a given letter in the acronym*, as illustrated by examples (2.10), (2.11), (2.80) or even the more extreme example (2.23), where the numeric information is *summarized.*

'3M' for "Minnesota Mining and Manufacturing Company" (2.10)

'C5R' for "Consortium of Canadian Centres for Clinical Cognitive Research" (2.11)

Rule 2.11 (Migration) *In some languages, similarity of pronounciation or similarity in the written form of characters in the expansion with characters in the acronym can be the only matching criterion. Generally, the matching character in the acronym is a* simplified form *of the matching character in the expansion.*

In languages where accents or other signs can be added to characters of the alphabet, characters in the expansion can *migrate* to unaccented counterparts in the acronym, such as in the French example (2.24), where 'É' in the expansion migrates to 'E' in the acronym.

Similarly, in languages with composite characters created from existing base characters using additional signs, such composite characters can migrate to the base character in the acronym. In Romanian 'Î' migrates to 'I' in example (2.51) and similar migrations occur from 'Ş' to 'S' and from 'Ţ' to 'T'. In Russian, 'и' migrates to 'й' due to phonetic constraints in example (2.59).

In Japanese, migration can occur from the use of Kanji in the expansion to Kana in the acronym, as in example (2.84).

Rule 2.12 (Inflection) *In languages with agglutinative morphology, group matches which represent whole morphemes can be inflected in the acronym.*

Example (2.60) illustrates the inflection to the genitive of one of the terms in multi-term syllabism in Russian.

Rule 2.13 (Consecutive matching) *Characters in the acronym match consecutively, in the same direction, terms (symbols, characters, words, expressions) in the expansion. Words skipped in the expansion are skipped in the same direction as the matches.*

Consecutive matching is the basis for dynamic programming algorithms for acronym acquisition, such as [132] and presented in chapter 4, but is not universally applicable to all domains or languages, as shown in the examples associated with rule 2.14 (Inversion) below. In a quantitative study presented in chapter 6 on eleven languages, examples of inversion are found for German, Finnish, Russian, and –with much lesser incidence– in English. English

Rule 2.14 (Inversion) *There are situations (exceptions to rule 2.13), where symbols in an acronym match terms in the expansion in reverse order.*

Acronym *inversion* in the English language is infrequent and its occurrences are debatable.

Hypothesis 2.4 (Syntactic Inversion) *Acronym inversion occurs in the abbreviation of expressions containing discontiguous overlapping syntactic constituents.*

One possible example (cited and disregarded as "not an acronym" by [132]) is represented in (2.12) where the order of the initials 'B' and 'F' is reversed. The symbolic match of "thousand" to 'M' must also be noted.

I submit that this occurrence is due to the existence of discontiguous overlapping syntactic constituents: "thousand feet" and "board measure", which, if rearranged to remove crossing lines in the parse tree (in figure 2.5) yields word order consistent with the order of matching letters in the acronym.

'MFBM' for "Thousand Board Feet Measure" (2.12)

Hypothesis 2.5 (Morphological inversion) *In languages with productive compound morphology, inversion occurs as a result of the difference of intra-word morpheme order and word phrase order for the abbreviated expression.*

Examples attributed here to morphological inversion such as (2.32) and (2.33) in German, are explained by the ordering of morphemes in possible larger compound words which describe the expansion, an ordering different than that of the words in the original phrase.

Example (2.81) is explained here through morphological inversion in Japanese, where the initial characters of words in a multi-word expressions also represent morphemes, which naturally combine

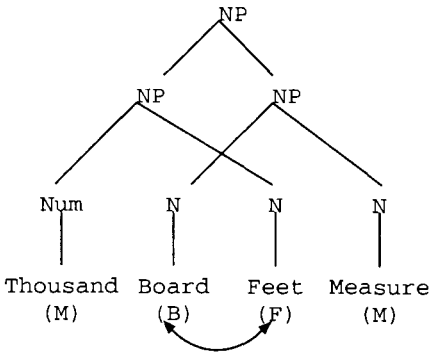


Figure 2.2: Possible parse tree with crossover for the expression "Thousand Board Feet Measure" (Figure by author)

into a *word-like* compound term. Inversion is attributed to constraints for the order of morphemes in the acronym.

Similarly, in Russian, morphological inversion occurs with syllabisms, such as in examples (2.61) and (2.62), where the acronyms behave as morphological compounds, being submitted to order constraints for intra-word morphemes, in this case attributes-to-head.

Hypothesis 2.6 (Accidental Inversion) *Acronym inversion occurs in situations where multiple alternate but semantically equivalent forms exist for a given abbreviated expression, and only one of them, not necessarily the most popular, is used to form an acronym. Preference is given, following hypothesis 2.2 to the expression with the highest pronounciability.*

Example (2.34) in German illustrates accidental inversion, where the most popular (and official) wording of the abbreviated expression is not the one used in the creation of the acronym.

Rule 2.15 (Import) *Acronyms can be directly imported from other languages, rather than created through abbreviation of a translated expansion.*

Acronym import is very common with technical acronyms, predominantly originating in the English language (such as 'HTTP', 'URL'), and with acronyms of US government organizations which have visibility beyond their borders. For example, the English language acronyms 'CIA' and 'FBI' are directly used (often without direct reference to their expansion) in languages such

as German, French, Spanish or Romanian. An estimated half of all entries in [73] are imported acronyms, most from the English.

Similarly, the acronym 'KGB' (2.52) is commonly used in English, without a need to know or specify its Russian expansion.

2.6 Cross-linguistic Perspective

Hypothesis 2.7 (Universality) *Acronyms are cross-linguistic phenomena and are present in the written form of most languages. Acronym formation rules are common, although their ordering and relative importance (weight) can be different from language to language.*

While it is not my objective here to give an exhaustive account of all acronyms in all languages, I will support this assumption by providing examples of acronyms in a number of languages with diverse grammars and writing systems. An exhaustive analysis of acronyms in *all* languages, an exercise which is left as future work.

Due to the exclusive English language coverage of all the reviewed literature on acronyms, the following discussion about acronyms in other languages is often focused on similarities and differences with English.

The investigative method in writing the following language-specific subsections was based on:

- Information available from hardcopy or online dictionaries of acronyms and abbreviations, mentioned whenever applicable.
- Online lists of acronyms. The multilingual list of acronyms from the European Union's Publication Office [36], which contains lists of acronyms and abbreviations in eleven languages was used whenever applicable.
- Interviews with native speakers.

2.6.1 Spanish

Spanish is a widely spoken language, with a variety of dialects.

Initial matching (rule 2.1) accounts for most acronyms (*acortamiento, sigla*) in Spanish: (2.13). Function word skipping (rule 2.6) is also common: (2.14) and (2.15). Morpheme matching (rule 2.2) occurs: (2.15), but with less frequency, since morphological compound words are only frequent in Spanish within certain domains, such as medicine.

Plural duplication (rule 2.9) is also present, if not widely used: (2.16).

'TLCAN' for "Tratado de Libre Comercio de América del Norte" (2.13)

'SICAV' for "sociedad de inversión de capital variable" (2.14)

'SIDA' for "síndrome de immunodeficiencia adquirida" (2.15)

'EEUU' for "Estados Unidos" (2.16)

Group matching (rule 2.4) is common in Spanish: (2.14), and is also used to combine proper names, especially within certain national Spanish-speaking groups (such as in the Republic of Cuba): (2.17), (2.18) and (2.19).

'Maribel' for "Maria Isabel" (2.17)

'Marisa' for "Maria Isabel" (2.18)

'Elián' for "Elizabeth Juan" (2.19)

2.6.2 French

Acronyms (*raccourcies, sigles*) are common in French. Two web sites claim 12,000, respectively 17,000 entries[3].

Acronym formation in French is very similar to English. Initial matching (rule 2.1) and function word skipping (rule 2.6), as illustrated by examples (2.20) and (2.21), account for the formation of most acronyms.

'ONU' for "L'Organisation des Nations Unies" (2.20)

'RDI' for "Le Réseau d'Information" (2.21)

Group matching (rule 2.4) is also common, as illustrated by examples (2.22), which leads many

[3]On the web http://www.translatum.gr/dictionaries/french-acronyms.htm (February 2004) contains a comprehensive list of web sites dedicated to acronyms and abbreviations in French

times to multi-syllable word-like constructions.

'ADIBIPUQ' for "Association des directeurs de bibliothèques publiques du Québec" (2.22)

Symbolic matching (rule 2.10, numeric multiplicative matching): example (2.23) and migration (rule 2.11): example (2.24) are also common occurrences.

'IN2P3' for "Institut national de physique nucléaire et de physique des particules" (2.23)

'EDF' for "Électricité de France" (2.24)

Duplication of the matching letters in the acronym (rule 2.9), to account for the plural form of the expansion (in this case royal), is also possible, although very rare:

'LLAARR' for "Leurs Altèsses Royales" (2.25)

2.6.3 German

Acronyms (*Akronyme*, *Abkürzungswort*) are common in German. There are a number of acronym dictionaries, including [73], which claims — in its title — over 50,000 abbreviations and symbols, although many of them are in English. Many online German acronym dictionaries and lists exist as well.

Initial matching (rule 2.1) accounts for most German acronyms: examples (2.26) and (2.27). Group matching (rule 2.4) is also present: (2.28) .

'ZDF' for "Zweites Deutsches Fernsehen" (2.26)

German has mandatory capitalization for all nouns (common and proper nouns), which leads in some cases to mixed capitalization of acronyms, as in examples (2.27) and (2.28).

'GmbH' for "Gesellschaft mit beschraenkter Haftung" (2.27)

'BfArM' for "Bundesinstitut für Arzneimittel und Medizinprodukte" (2.28)

The German language has very rich morphology, especially productive through the phenomenon

of "compound nouns". As a result, acronym formation through the application of rule 2.2 is common, sometimes repeatedly for the same word in the expansion, as in (2.29), (2.30) and (2.31).

'KGI' for "Katholische Glaubensinformation" (2.29)

'PLZ' for "Postleitzahl" (2.30)

'KFZ' for "Kraftfahrzeug" (2.31)

Morphological inversion (rule 2.14, hypothesis 2.5) occurs in German, as illustrated by examples (2.32) and (2.33).

I submit here that multi-word expressions in German can be abbreviated considering morphological ordering of their component terms, *as if they formed large compound words* (sometimes exaggerated). To illustrate, one possible way of rephrasing the expansion in example (2.32) is "Europäisches Fernsehsendungsschutzabkommen", where the second word, although grammatically correct, is frowned upon as *exaggerated* or *overly bureaucratic* by native speakers. The order of letters found in the acronym matches in this situation the order of letters in this reworded expression, which follows the regular attributes-to-head intraword morpheme order in German.

'EFA' for "Europäisches Abkommen zum Schutz von Fernsehsendungen" (2.32)

'BWÜ' for 'übereinkommen ber biologische und Toxinwaffen' (2.33)

Example (2.34) represents an occurrence of accidental inversion (rule 2.14, hypothesis 2.6). An alternate expression: "Europaeisches Gericht Erster Instanz", semantically equivalent to the *official* and most frequently used expression which is the expansion is used in the creation of the acronym, resulting an acronym with increased *pronounciability*: 'EuGeI', compared to the 'GEIEu', should the original expression have been matched consecutively, as specified by rule 2.13.

'EuGeI' for "Gericht Erster Instanz der Europaeischen Gemeinschaften" (2.34)

2.6.4 Finnish

Finnish is a language with very productive agglutinative morphology [31], where compound and inflected words such as (A.2), with parse tree in Figure A.1(a) are common.

This leads to frequent morpheme matching for acronyms, such as in examples (2.35) and (2.36). Since most function words which appear in the translation of expansions in another language (e.g.

English) are included in compound terms in Finnish, the application of function word skipping (rule 2.6) is greatly reduced, compared to English.

'YTK' for "yhteinen tutkimuskeskus" (2.35)

'TKK' for "teollisuuden kehittämiskeskus" (2.36)

It is submitted here that acronym inversion (rule 2.14, hypothesis 2.5) is systematic in Finnish, similarly with German, given restrictons on morpheme order in compounds (head-to-attributes), which follow in the acronym.

'BKTmh' for "markkinahintainen bruttokansantuote" (2.37)

2.6.5 Italian

Acronyms (*abbreviamento*) are common in Italian[4].

Initial matching (rule 2.1) accounts for the the formation of most acronyms, such as in example (2.42). Function word skipping (rule 2.6) is also very productive: examples (2.38) and (2.40).

Intra-word morpheme matching (rule 2.2) is also present: examples (2.38) and (2.39). The aplication of group matching (rule 2.4): examples (2.40) and (2.41) can result many times in multi-syllable word-like expressions.

'SIDA' for "Sindrome da Immunodeficienza Acquisita" (2.38)

'RCA' for "Republica Centrafricana" (2.39)

'ANASIN' for "Associazione Nazionale Società di Informatica" (2.40)

'SIDARMA' for "Società Italiana Di Armamento" (2.41)

Mixed capitalization, following the capitalization of individual words in the expansion many times leads to mixed case acronyms, such as (2.42).

'S.p.A.' for "Società per Azioni" (2.42)

[4]On the web http://parole.virgilio.it/parole/abbreviazioni_e_sigle (February 2004) links to a web dictionary for italian acronyms

2.6.6 Hungarian

Hungarian is another language with agglutinative morphology. Most acronyms (*betüszó*) match on initials (rule 2.1), as in example (2.43). Function word skipping (rule 2.6) is present: examples (2.44) and (2.45), but its occurrences are reduced, due to the existence of suffixes that play the role of most prepositions in languages such as English.

'MTA' for "Magyar Tudományos Akadémia" (2.43)

Group matching (rule 2.4) occurs frequently in Hungarian, as in examples (2.47) and the group 'ma' in example (2.44), but has also occurs systematically for the letter group 'sz' (pronounced **s**), which is generally treated as one single letter, as in examples (2.44), (2.46), (2.48).

'MAFSZ' for "Magyar Független Film és Video Szövetség" (2.44)

'SZTAKI' for "Számìtástechnikai és Automatizálási Kutató Intézet" (2.45)

'MSZTT' for "Magyar Szellemi Tulajdonvédelmi Tanács" (2.46)

'MÁÉRT' for "Magyar Állandó Értekezlet" (2.47)

'KRESZ' for "Közlekedésrendészeti Szabályok" (2.48)

Accented letters in Hungarian represent generally (with few exceptions) different sounds than their unaccented counterparts and maintain their accented form in the acronym, as in example (2.47). Examples of migration in Hungarian are not known to the author.

Morphological compound words are common in Hungarian, and so are occurrences of morpheme matching (rule 2.2), as in examples (2.45) and (2.48).

2.6.7 Romanian

Initial matching (rule 2.1) and function word skipping (rule 2.6) account for the formation of most acronyms (*acronim*) in Romanian: examples (2.49) and (2.50). Migration: 'Î' migrates to 'I', as in example (2.51), from 'Ş' to 'S' and from 'Ţ' to 'T' is also common.

'PNŢCD' for "Partidul Naţional Ţărănesc Creştin-Democrat" (2.49)

'ICI' for "Institutul de Cercetari în Informatică" (2.50)

'ITA' for "Întreprinderea de Transporturi Auto" (2.51)

2.6.8 Russian

Russian is a Slavic language, which uses the cyrillic alphabet, similarly with Bulgarian, Ukrainian and Serbian. A number of other Slavic languages (such as Czech, Slovac, Polish) have converted to Latin alphabets.

Russian has a significant number of acronyms, as illustrated by [121], containing "about 40,000 abbreviations", counted as acronym forms, with the number of acronym expansions contained therein estimated to exceed 100,000.

Most acronyms in Russian are also formed through initial matching: (2.52), (2.53). Due to very productive noun inflection (e.g *genitive*) and the lack of definite (*the*) and indefinite (*a*) articles, function word skipping (rule 2.6) is present, although significantly less frequent than in English: (2.53).

'КГБ' for "Коммитет Государственной Безопасности" (2.52)

'МБДП' for "Министерство
"Бумажной и Деревообрабатывающей Промышленности" (2.53)

Repeated applications of group matching (rule 2.4) lead to the creation of syllabisms, such as (2.55), (2.56), and sometimes multi-syllabism expressions, such as in (2.57) phenomena which were especially productive during the Soviet era, used for multi-level hierarchical organization names (e.g. Institute ... of the Regional Administration ... of the Ministry of ... of the Soviet Socialist Union).

Proper noun capitalization rather than *all capitals* is commonly used with syllabisms in Russian, such as in examples (2.57), (2.59), , (2.60), (2.61), (2.62). Many acronyms, such as (2.55) or (2.56) were frequently used during the Soviet era, in the mainstream body of language, losing their (possible) original capitalization, in a similar fashion with the English 'laser' or 'radar'.

Ambiguous capitalization of acronyms poses significant challenges to their automated acquisition. I submit here that this is also of lesser importance, given that capitalized acronyms in Russian, as described above, fit definition 2.9 (*proper names*), and uncapitalized ones are migrated to common words, to be found in general-purpose dictionaries, as for example (2.55).

'ДГЭС' for "Днепровская Гидроэлектростанжия" (2.54)

'комсомол' for ' "Ленинский комунистически союз молодежей" (2.55)

'леспромхоз' for " Лесопромышленное Хозяйство" (2.56)

'Омскснаб Запсибчерметснабсбыта' for "Омская государственная контора материально- технического снабжения Западно-Сибирского управления материально- технического снабжениа и сбыта продукции черной металлургии Главного управления металлов Государственного комитета материально- технического снабжения и сбыта " (2.57)

Russian has very productive compound morphology, leading to frequent applications of rule 2.2, (2.58):

'АДО' for "Авиадесантны отряд" (2.58)

Migration (rule 2.11) occurs in cases such as (2.59) and (2.60) from 'и' to 'й'. Inflection (rule 2.12) occurs in cases such as (2.60), in this example with the noun-like inflection in the Genitive case of the second term of the syllabism, from 'Минстрой' to 'Минстроя' .

'Кандалакштрансстрой' for "Кандалакшиинский трест транспортного строителства" (2.59)

'Главсибстрой Минстроя' for "Главное управление по строительству в районах Сибири Министерства строительства" (2.60)

Morphological inversion (rule 2.14) appears to occur systematically in Russian, examples (2.61) and (2.62), for syllabisms where whole morphemes are retained from words in the expansion, forming novel morphological compounds, which must conform to intra-word morpheme ordering attributes-to-head.

'Гидростройпроект' for "Государственный институт
построительству и рабочемупроектированию гидроэнергоузлов" (2.61)

'Рыбсбыт' for "Управление сбыта рыбы и рыбопродуктов" (2.62)

2.6.9 Bulgarian

Bulgarian is a Slavic language, related to Russian, and with similar acronym production regularities, based mainly on initial matching (2.63), and also on rich compound morphology, resulting in morpheme matching, such as in (2.64). Compared to Russian, Bulgarian has less productive noun inflections, more frequent use of function words, and consequently, function word skipping (2.63). The expansion in this example, if translated to Russian, maintains the word order in the phrase, but loses the function word "на" (of), in favor of the declination of the noun "Армиа" in the genitive case: "Армии".

'ЦСКА' for "Централен Спортен Клуб на Армиата" (2.63)

'ВМА' for "Военномедицинска Академия" (2.64)

2.6.10 Hebrew

Hebrew is an ancient Semitic language [117] which has survived mainly through religious usage and writings, was revived around the turn of the twentieth century and is now spoken and written in the state of Israel and by Jewish populations around the world. The written form of Hebrew[5] is based on an alphabet (*alefbet*), with a number of *quiet* symbols (written but not pronounced), and a number of *unwritten* vowels. Some characters have multiple pronounciations, depending on context.

Abbreviations are used in conjunction with the *Passover Haggadah*, a biblical narrative, as mnemonics for the ten plagues. The acronym in example (2.65), phonetically transcribed 'D'TZaKH',

[5]A brief account of the Hebrew writing system by Tracey R. Rich can be found on the web at http://www.jewfaq.org/alephbet.htm (February 2004).

for "Dam TZ'fardea Kinim", translated as "water-turning-to-blood, frogs, vermin", which encompasses the first three biblical plagues.

דצ"ך for דם צפרדע כנים (2.65)

Since Hebrew does not use capitalization, there is no clear differentiation based on capitalization between *acronyms* and *direct abbreviations*, as defined in section 2.4. Hebrew abbreviations (*rashey tevot*) are indicated through the use of " before the last character in the abbreviation, and are generally *not initialisms*, but are *vocalized* as new, stand-alone words, through the addition of unwritten vowels, usually **a**.

An online list of acronyms in Hebrew[6] contains about 250 entries.

Acronym formation in Hebrew is based mainly on initial matching (rule 2.1), as in examples (2.66): 'SHaH' , an acronym for Israel's currency "Shekel Hadash" (New Israely Shekel) and (2.67): 'SHaBaK', an acronym for "Sherut Bitakhon Klali", translated as "Central Security Service". Occurrences of 'SH' in this example stand for one letter ש, and the vowel 'a' is not written, but pronounced.

ש"ח for שכל חדש (2.66)

שב"ך for שרת ביתכן כלל (2.67)

Group matching (rule 2.4) is also common, as in examples (2.68): 'MaGaV', an acronym for "Mishmar Ha-Gvul" translated as "Frontier Guard" (a combat branch of the Israeli Police).

מג"ו for מישמר חגול (2.68)

Hebrew also supports initialisms, acronyms read through the consecutive pronounciation of their initial letters, as illustrated by example (2.69): 'Mem-Pey' for "Mefaked Pluga" (company commander)

מ"פ for מפכד פלג (2.69)

[6]On the web at http://www.stands4.com/browsecategory_R.asp?CATEGORY=HEBREW&page=1 (February 2004).

2.6.11 Arabic

Arabic is a widely used language, with a variety of spoken forms. Standard Arabic [9] is spoken in educated environments and also represents the written form of Arabic, evolved from classical texts. Arabic has a writing system [54] close to *calligraphic hand writing*, which uses extensively cursive links and even transformations between adjoining symbols. All letters have corresponding sounds, and there are a number of *unwritten* short vowels (pronounced but not written). Supplementary to the common alphabet (Modern Standard Arabic), a number of symbols occur only in certain geographic regions.

Acronym formation is based on similar rules as in languages with Latin alphabets. The following example[7] (2.70) illustrates initial matching (rule 2.1) for acronym transcribed as 'HAMAS', with the expansion transcribed from Arabic as "Harakat Al-Muqawama Al-Islamia", translated as "Islamic Resistance Movement". 'Hamas' is also a word, meaning courage and bravery.

حماس for حرکت المقاومه الاسلامیه (2.70)

2.6.12 Farsi

Persian [127] is a group of languages from the Indo-Iranian family, with two major dialects: Farsi (in Iran) and Dari (in Afghanistan). Farsi is the official and the most widely-spoken language of Iran.

Persian[8] writing in Iran and Afghanistan uses a variety of the Arabic writing system, called Perso-Arabic.

Acronyms formation in Farsi is based on initial matching (rule 2.1), as illustrated by the following examples, transcribed in Latin, with the addition of vowels in small capitals that are pronounced, but not written. Transcription also induces deviation from the exact matches in the original script, for 'I' and respectively 'E' in the expansion to 'A' in the acronym.

'HoMA' (2.71) is an acronym for "Havapeimaii Melli Iran", translated "Iran Air". 'PeKA' (2.72) is an acronym for "Pakhsh Ketab Iran", translated as the "Iranian Book Distribution Company". 'NAJA' (2.73) is an acronym for "Nirooye Entezami Jomhoori Eslami", translated as the "Islamic Republic Police Force".

[7] Attested on the web at http://www.globalsecurity.org/military/world/para/hamas.htm (February 2004).

[8] A profile of the Persian language can be found on the web http://www.lmp.ucla.edu/profiles/profp01.htm (February 2004).

هما for هواپیمایی ملی ایران (2.71)

پکا for پخش کتاب ایران (2.72)

ناجا for نیروی انتظامی جمهوری اسلامی (2.73)

2.6.13 Chinese

Chinese is a widely spoken language, with numerous dialects, some as far apart as distinct languages, unified by a common, multi-millenary writing system. Chinese has a large number of characters: 56,000 attested forms, of which 6,600 provide coverage of 99.999% of modern published material [24]. Completely separate languages, Korean and Japanese, have also imported traditional Chinese ideograms during past centuries.

Chinese ideograms represent syllables, although their pronounciation can be different within different geographical regions, and multiple characters can be pronounced similarly. [24] accounts for 137 different characters that are pronounced *ji*. A relatively large number, 46% of words in circulation in modern Chinese are monosyllabic, and most morphemes, 87.5% are monosyllabic.

Acronyms in Chinese fall into the definition of syllabisms. Most are formed through initial matching (rule 2.1) and word skipping (rule 2.8), as illustrated by the following examples, transcribed in *Hányu pinyin* (phonetic transcription in roman alphabet).

'Gushi' (2.74), an acronym for "Gupiao Shichang" is translated as "stock market". 'Nv Zu' (2.75) for "Nv Zi Zuqiu" is translated "women's soccer". 'Cai Dian' (2.76), for "Cai Se "Dianshiji" is translated "color TV".

股	市	for	股	票	|	市	场	(2.74)
gu	shi		gu	piao	|	shi	chang	
女	足	for	女	子	|	足	球	(2.75)
nv	zu		nv	zi	|	zu	qiu	
彩	电	for	彩	|	色	|	电	视	机	(2.76)
cai	dian		cai	|	se	|	dian	shi	ji	

Morpheme matching (rule 2.2) is illustrated by 'Beiyue' (2.77), an acronym for "Bei DaXiYang Gongyue Zuzhi", a translation in Chinese of the "North Atlantic Treaty Organization" (NATO).

北	约	for	北	大	西	洋	公	约	组	织
bei	yue		bei	da	xi	yang	gong	yue	zu	zhi

(2.77)

2.6.14 Japanese

Japanese phonology is based on a sub-syllabic smallest phonetic unit: the mora. Japanese [23] has three different writing systems, and one Latin transcription system:

- *Hiragana*, a traditional writing system, where symbols have phonetic (mora) values. Hiragana is used for "phonetic" writing (e.g. by children or by foreigners), for writing prepositions, adverbs and for applying inflection (adjective, verb terminations).
- *Kanji*, a traditional writing system, imported from China. Kanji characters represent morphemes with semantic value, and –most of them– have multiple pronounciations, depending on context. Kanji symbols represent base concepts, which, many times, combine into so called "compound concepts".
- *Katakana*, a writing system used mainly for the Japanese transcription of so called "loanwords", i.e. foreign (Western) words.
- *Romaji*, a system of phonetic transcription using Latin characters.

Hiragana, Kanji and Katakana are commonly mixed in written text[9].

Japanese acronyms[10] are systematic abbreviations above this written complexity. Since each Japanese character encodes at least one mora, all Japanese acronyms can be interpreted as syllabisms.

Initial matching (rule 2.1) and word skipping (rule 2.8) account for most Japanese acronyms, through the retention into the acronym of the initial character of some of the words in the expansion, in consecutive order, as illustrated by the following examples.

'Ken-Po-Ren' (2.78), for "Kenko Hoken Kumiai Rengo Kai", is translated to "National Federation of Health Insurance Societies". In the context of the pronounciation of the acronym, the initial syllable of "Hoken" is, pronounced and transcribed in roma-ji as 'Po'.

[9]The web site http://www.kanjisite.com/index.html (February 2004) provides an excellent brief introduction to the Japanese writing systems.

[10]The web site http://www.inv.co.jp/~yoshio/DW/Ryaku/RyakuA.htm (February 2004) has been used as source for the following examples of Japanese acronyms.

'In-Ten' (2.79), for "Nihon Bihutsu In Tenrankai" is translated to "Japan Fine Arts Exhibition".

健	保	連	for	健	康	保	険	組	合	連	合	会	、
ken	po	ren	for	ken	kō	ho	ken	ku	miai	ren	gō	kai	(2.78)
院	展		for	日	本	美	術	院		展	覧	会	
in	ten		for	ni	hon	bi	jutsu	in		ten	ran	kai	(2.79)

Symbolic matching (rule 2.10, numeric multiplicative matching) is also present in Japanese as in example (2.80), 'San Ko', where the initial syllable-morpheme and word ('ko', translated to 'high') is identical for all three abbreviated phrases in the expression: "Ko ...", and is translated to "high educational background, high body weight, high income". 'Three high' is also a literal translation of the acronym.

三	高	for	高	学	歴	・	高	身	長	・	高	収	入
san	kō	for	kō	gaku	reki		kō	shin	chō		kō	shū	nyū	(2.80)

Acronym inversion (rule 2.14) occurs systematically in Japanese, as in example (2.81) 'noshi-rin-cho', for "Rinji Noshi Chousa Kai" (meaning "Emergency Committee for Transplant from Braindead Donors") due to differences between the constraints for word order in the phrase and morpheme order in the resulting acronym, which should have attributes before head phrases. In this situation, 'noshi' (translated "braindeath"), acts as a head to the attribute to 'rin', from "rinji" (translated "emergency").

脳	死	臨	調	for	臨	時	脳	死	及	び	臓	器	移	植	調	査	会
nō	shi	rin	chō	for	rin	ji	nō	shi	oyo	bi	zō	ki	isho	ku	chō	sa	kai	(2.81)

Import (rule 2.15) through Katakana transcription of the pronounciation of an English acronym is also a fairly common phenomenon, as illustrated by example (2.82), where 'JETRO', an English language acronym for "Japan External Trade Organization" is phonetically transcribed in Katakana. The pronounciation or writing of the transcribed foreign acronym has no relationship to the pronounciation or writing of the Japanese expansion.

ジ ェ ト ロ
je to ro for "Japan External Trade Organization" (2.82)

A similar phenomenon of import is illustrated by 'PokeMon', an indirect (transcribed from Katakana) acronym for "Pocket Monster", which phonetically transcribes in English the Katakana acronym for the Katakana phonetical transcription in Japanese of "Poketto Monsutaa" (Romaji).

ポ ケ モ ン for ポ ケ ッ ト | モ ン ス タ ー 、
po ke mo n po ke t to | mo n su tā (2.83)

Finally, example 2.84 'Karaoke' is an acronym for the mixed Kanji-Katakana expression "Kara O (empty song) okesutora (orchestra)", with meaning "Orchestra without Song". The acronym is completely written in Katakana, including the part written in traditional Japanese Kanji in the expansion, which illustrates acronym migration (rule 2.11).

カ ラ オ ケ for 空 っ ぽ の | オ ー ケ ス ト ラ
ka ra ō ke kara p po no | ō ke su to ra (2.84)

2.7 Conclusions

The construction, analysis and evaluation of systems which collect, normalize or use abbreviations and acronyms has been hindered by the lack of a consistent terminology, which characterizes entities *by what they are*, and provides adequate coverage in a variety of domains and languages.

Definitions for acronyms, abbreviations and related terms are provided here, and the relationships between acronyms and related terms, such as non-standard words, direct abbreviations, names and creative spellings are discussed. Differentiation tests are proposed for terms which are ambiguous with acronyms.

A universal explanatory theory for acronym formation is proposed, which rests on seven linguistically testable hypotheses (axioms). A set of fifteen universal violable rules is proposed, which is partly redundant and contradictory, and attempts to explain the formation of all acronyms in all languages.

Justification from hypotheses and examples for all rules are provided, applications and particularities in fifteen different languages are discussed. While testing of the hypotheses has been executed beyond the provided examples, it is also left as a future ongoing effort, for other languages, or until contradictory or new, uncovered examples are found.

An ordering for the rules for acronym formation is provided further, in chapters 3, 4 and 5, which results in usable acronym acquisition systems. A quantitative study, based on a corpus of experimental data, which provides specific orderings or weights for twelve different languages, is presented in chapter 6.

The next chapter (3), presented as a stand-alone article, describes a greedy algorithm for acronym acquisition. The algorithm identifies acronym definitions and cases of cooccurrence of acronyms with their expansions by attempting successive applications of a subset of the acronym matching rules presented in section 2.5, in an specific order, for the English language. If a match has been obtained for a higher-order rule, the algorithm moves forward, and backtracks if a complete match cannot be found.

Chapter 4 uses a similar rule ordering for English and chapter 6 presents orderings for ten other languages. A specific scale of weights is also introduced, for each language, for use in a dynamic programming algorithm (similar with string alignment), tries to maximize a matching score, calculated as the sum of the weights of all rules used to match an acronym to its potential expansion.

Chapter 3

A Linguistic Approach to Extracting Acronym Expansions from Text

Reproduced from *Knowledge and Information Systems volume 6 issue 3* with permission.

ABSTRACT

We propose a linguistically-aided approach for the extraction of acronyms from text, with implications at the discourse, lexicon, morphologic and phonologic levels. A system implemented using this approach achieves excellent performance (f=95.22%) on a limited corpus. Our work indicates the adequacy of linguistic methods for acronym discovery.

3.1 Introduction

An acronym is a "word formed from the initials of other words" cf. [7]. In this work, we use a more extensive definition of acronyms as abbreviations of complex, possibly multi-word expressions, through mechanisms based on the association of some letters in the acronym with some of the words in the expression. We refer to the expression originating an acronym as the "expansion" of the acronym, and to expressions containing both an acronym and its expansion with the purpose of defining the meaning (expansion) of the acronym as "acronym definitions". We use a simple definition of acronym candidates, following capitalization patterns and allowing for intra-acronym

punctuation, as in the examples RFC, U.S.A., DoD, TCP/IP, S-MIME, RFCs, MIB-s or RR's. We exclude acronyms containing digits and other non-alphanumeric characters, as in 3M or MP+.

[8] attempt the automatic discovery of acronym expansions by proposing a method based on inexact pattern matching applied to text surrounding the possible acronym, through maximization of a confidence function related to an inexact match between the first letter in words in expansion candidates and letters in the acronym. Their system — Acronym Finding Program (AFP) achieves 93% recall and 98% precision on text from a restricted domain: government studies relevant to the Yucca Mountain Waste Disposal Project. Their approach sacrifices potentially important information, contained by letters inside words in the expansion, and compensates that by not taking into consideration some more difficult expansions, such as DOP for "dioctyphthalate".

[10] uses a heuristic-based system called TLA (Three Letter Acronym), and a compression-based method ([11]) that suggests similarities with statistical collocation discovery approaches.

[5] uses combinations between "canonical" and "contextual" approaches in a spectrum of 4 different algorithms. The canonical algorithm attempts to match acronym definitions by looking for definition patterns, such as "Expansion (Acronym)", as in the example: "Department of Defense (DoD)"

The contextual algorithm attempts to match prefixes and up to 4 letters from all words in the expansion with consecutive letters in the acronym. The highest performance (measuring $f = precision \times recall$, 92% precision and a 88% recall) is achieved considering acronyms longer than 2 letters and excluding distances longer than 20 words between an acronym and its expansion.

A number of manually built dictionaries of acronyms, including the World Wide Web Acronym and Abbreviation Server [9] and Acronym Finder [1], containing extensive lists of acronyms from many domains, are sources likely to provide Internet users with the most significant coverage of general purpose acronym searches. [6] provides a good starting point.

We have not identified any prior acronym expansion algorithms with a linguistic motivation or methodology, which is surprising, as acronyms are linguistic phenomena. In the description of their contextual algorithm, [5] note that it includes "a crude morphemic decomposition, but without any knowledge of English prefixes". Why not go further ?

3.2 Methodology

We propose a linguistic corpus-based approach for the automatic discovery of acronym expansions, that enhances aspects of the reviewed methodologies. Linguistic information is used at the levels of

discourse (by limiting searches to paragraphs of text), lexicon, morphology and phonology.

Our algorithm operates in 2 successive passes, a definition matching pass and an expansion matching pass. The expansions resulting from both passes are compounded into a single list. The recall of the two passes is compounded, and precision shows also a slight increase for the test data.

During the definition matching pass, processing starts from every occurrence of an acronym, and matching the pattern of a definition is attempted. This may result in expansion candidates that are located forward or backward from the occurrence of the acronym. The method for matching definition candidates is somewhat similar with the one used by the "Canonical/Contextual" algorithm described in [5], and uses an extensible library of definition templates.

Matching of letters in the acronym with letters from the expansion is attempted based on a number of non-strict acronym matching rules that are tried successively, in different order for backward and forward matching:

- Initial match: the initial of the word is matched with the current letter of the acronym.
- Noise words are ignored: a list of noise words is built from words that can form articles, prepositions or conjunctions.
- Subacronym match: a complete acronym can be incorporated into another acronym; e.g. "VLAN" for "virtual LAN".
- Morphological prefix match: terms in a list of English language prefixes are matched with the prefix of the current word in the expansion; if successful, both the initial of the prefix and of the remainder of the word are considered part of the acronym; e.g. "hypertext" matches "HT", the beginning of "HTML", an acronym for "Hypertext Markup Language".
- Prefix match: letters from the beginning of a word in the expansion are identical with the current group of letters of the acronym; e.g. the first 4 letters of the word "bootstrap" match "BOOT", at the beginning of the acronym "BOOTP", for "Bootstrap Protocol".
- Syllable-driven match: first letter(s), or first consonant(s) in every syllable are matched with current letters of the acronym; e.g. the word "connectionless" is broken into syllables con-nec-ti-on-less, and the initials of syllables "con" and "less" match "CL" from the beginning of CLNP, an acronym for "Connectionless Network Protocol".
- Word omission: words introduced by punctuation "/" or "-" may be omitted in matching

letters in the acronym; e.g. the word "Level" is omitted in the expansion "High-Level Data Link Control" of the acronym "HDLC".

- X: special consideration is given to matches involving the letter "X" encountered in an acronym, which is also considered to match "ex" in the beginning of words in the expansion, as in "XML", and acronym for "Extensible Markup Language". This rule may only be specific to the domain covered by our corpus.

The first expansion matching an acronym is considered valid and the algorithm moves onto the next definition candidate. Syllable matching is handled using Mike Hammond's "Locally Encoded Syllable Parser" ([3], [4]), which is incorporated as a module.

The expansion matching pass attempts to match only the initials of words, including the substitution of 'X' for initial 'ex' similar with the last rule from the definition matching phase. The algorithm is only applied to paragraphs containing acronyms without expansions identified during the definition matching pass. Noise words or words introduced by punctuation "-" or "/" may also be omitted from possible expansions.

Only acronyms with lengths more than 2 characters are considered by the expansion matching algorithm, the probability of accidental matches for random pairs of consecutive words being considered too high (average approx. 1.5%).

The expansion matching algorithm is somewhat similar to, even if much simpler than the algorithm in [8] or the Contextual algorithm in [5]. Our most important contribution here is represented by the use of the expansion matching pass only for acronyms unmatched by the definition matching pass.

3.3 Discussion

The need for both backward and forward matching is justified by our greedy approach, as stop conditions (conditions in which a specific matching path is abandoned) for recursive backward acronym definitions cannot be achieved without attempting to match initially the last letters of the acronym, progressing toward the beginning of the text.

In the following acronym definition: "The Internet Security Association and Key Management Protocol (ISAKMP)", after matching "KMP" at the end of the acronym candidate with "Key Management Protocol", the following 'A' from the acronym candidate is matched to the initial of 'and'.

'S' can be further matched to the first consonant of the initial syllable ('as') of the word 'association'. The following 'I' cannot be matched and the current matching assumption fails, back to 'and', which is found to be in the list of noise words and can be ignored, the match continuing successfully toward matching 'ISA' with "Internet Security Association".

In the previous example, if only a forward matching algorithm would have been available, ISAKMP would have had to be matched successfully, working backward from "Protocol", "Management Protocol", until the successful match has been identified. This method is not only less efficient, but could also lack an objective termination condition for expressions that are not acronym definitions, unless a test similar to the backward-matching algorithm is used to determine if an expression can be the last part of an expansion for the sought acronym.

In the example: "This document describes an application programming interface and a corresponding protocol (MGCP)", a backward pass would match the final 'CP' to "corresponding protocol", recursively ignore the noise words 'and' and 'a' and fail afterward. Successive forward matching passes for "protocol", "corresponding protocol", etc. would prove unsuccessful, without an intrinsic termination of the execution.

3.4 Evaluation

Our system is implemented as a series of Prolog modules, glued together by Perl scripts, that also serve for pre-processing of input and formatting of output. We evaluate performance using a corpus built from the initial paragraph of all abstracts of the Internet Engineering Task Force (IETF) Request for Comments (RFC), submitted before the middle of November 2001.

The Internet Requests for Comments (RFCs) are a component of the Internet standards process, and are, cf. [2], "concerned with all protocols, procedures, and conventions that are used in or by the Internet". Around the middle of November 2001, the IETF repository contained 2703 RFCs, with creation dates ranging from 1969 onwards.

We separate the abstract of each applicable RFC, resulting in a number of 1,470 abstracts, totaling 83,505 words, and representing around 576 KB of text (the Complete Corpus). One hundred documents are chosen at random from the first 1,277 documents, for analysis and the development of the system (Development Corpus). The last 193 documents were set aside for testing and never seen during the development stage (Test Corpus).

The complete corpus contains 681 different acronym candidates, occurring in 4735 separate instances. The maximum number of occurrences of an individual acronym is 443 (for RFC, an

acronym for "Request for Comments"). The execution of a run on the complete corpus of 1470 abstracts (approximately 576 KB) took approximately 25 minutes on a system with a Pentium II, 200 MHz processor, running Microsoft Windows NT 4.0.

A typical measure used in the evaluation of information extraction systems is $f = precision \times recall$. We suggest that in the context of our problem, there are at least 2 possible sets of definitions of precision and recall:

- Expansions: precision and recall are calculated starting from acronym expansions. This method is used by [5] and seems to be used for the evaluations of all the other systems reviewed. We present an evaluation using this definition for positioning our results within the rest of the work in the area.

  $$precision = \frac{identified:correct:expansions}{identified:expansions}$$

  $$recall = \frac{identified:correct:expansions}{total:expansions}$$

- Occurrences: precision and recall are normalized across the number of occurrences of each acronym. We suggest that evaluation using this definition is a more adequate measure of the usefulness of a certain algorithm on a given test corpus and domain. We present the results of the evaluation using this definition for future reference.

  $$precision = \frac{occurrences:of:acronyms:with:correct:expansions}{occurrences:of:acronyms:with:identified:expansions}$$

  $$recall = \frac{occurrences:of:acronyms:with:correct:expansions}{occurrences:of:acronyms:with:expansions}$$

Precision is calculated by manually verifying the correctness of each expansion. Recall is calculated by manually verifying whether acronyms occurring in a certain abstract have unmatched expansions within that abstract.

For the occurrence precision, the impact of incorrect expansions for acronyms with multiple expansions is normalized across all expansions. For example the acronym "TCP/IP" has 1 correct expansion, 1 incorrect expansion, and 70 occurrences. 35 uses of the acronym with incorrect expansions will be counted toward the precision ($35 = \frac{1}{2} \times 70$).

We measured the performance of the system separately for the Test Corpus and for the Complete Corpus (including test data). The experimental results are displayed in Table 1.

We encountered 7 incorrect expansions, a number of them due to greedy matches, using the omission of a word, by either finding it in the list of noise words (the prepositions "plus" or "for"), or by associating it with punctuation "-". The following incorrect expansions fall into this category:

		definition match			expansion match		
		precision	recall	f	precision	recall	f
test corpus	expansions	95.52%	93.01%	88.84%	95.69%	96.94%	92.76%
	occurrences	94.11%	93.32%	87.82%	94.23%	95.35%	89.84%
complete corpus	expansions	98.58%	93.19%	91.86%	98.63%	96.54%	95.22%
	occurrences	97.09%	94.16%	91.52%	97.14%	95.98%	93.23%

Table 3.1: Precision and recall of acronym expansion discovery for the corpus of RFC abstracts.

- TCP/IP is matched to "Transmission Control Protocol/Internet Protocol-based".
- MP is matched to "Multilink Protocol Plus". The real acronym in this situation is "MP+", which we did not consider.
- LDP is matched to "for Label Distribution Protocol"

The acronym RSVP was incorrectly matched to "Reservation Protocol", rather than "Resource Reservation Protocol", by our greedy matching algorithm, also due to our syllable-matching rule that can never omit the initial syllable of a word.

The acronym IP was incorrectly matched to "IPsec", as part of an acronym definition for IPsec, an expression that we did not consider an acronym candidate: "IPsec (IP Security)". Arguably incorrect expansions are found for FRF ("Frame Relay Function" rather than only "Frame Relay Forum") and VCI ("Virtual Connection Identifier") during the expansion matching pass.

Our system did not find 31 different valid expansions, mainly due to the relative weakness of our expansion matching pass for acronyms that did not occur within a definition. Three 2-letter acronyms ("OS" for "Operating System","US" for "United States" and "DH", for "Diffie-Helman") were not matched, being deliberately ignored in the expansion-matching pass.

Some more complicated expansions, which would have been caught by the definition matching phase (if included in definitions), were overlooked during the expansion matching pass; e.g. MARC for "Machine-Readable Cataloging".

A number of expansions with changed order were not caught, such as WG-MSG for "Mail and Messaging working group". Acronyms that omit completely elements of a word were deliberately not caught; e.g. VLSP for "Virtual LAN Link State Protocol".

Of interest was the acronym IXFR that we could not match to "incremental zone transfer", where the letter X is substituted for the prefix "trans". Intuitively, similar substitutions may also occur with other prefixes lacking an X, such as "cross". A puzzling acronym expansion we could not identify

is IKE for "automated KEY exchange", maybe created through phoneme similarity, or possibly a typing error.

Our evaluation does not exclude 2-letter acronyms, even if we have found marginal improvement when doing so. From this point of view, our evaluation is significantly more stringent than the ones in the reviewed literature; the results in [5] and [10] are based on 3-letter or longer acronyms.

3.5 Conclusions

Our system exhibits excellent performance for the Complete Corpus as well as for the Test Corpus, both from the point of view of the precision and the recall.

The figures obtained after matching definitions only (98.58% precision and 93.19% recall) are superior to the ones cited in the reviewed literature, when comparing f. After the matching expansion pass, recall is significantly improved beyond its previous value, with a marginal improvement in precision.

We believe that the performance measured on the smaller unseen Test Corpus is consistent with, even if lower than, the performance measured on the Complete Corpus ($f = 92.76\%$ vs. $f = 95.2\%$), which seems to promise an encouraging level of robustness of the algorithm. Discrepancies can be associated with the small size of the Test Corpus.

Even if testing on different corpora is needed for proving the generalizability of the results, we believe that our work clearly indicates the adequacy of using linguistic methods for discovering acronym expansions. We have not been able to perform a direct comparison of our algorithm to the ones cited in the literature because of the lack of an available common corpus. The evaluation of [5] is performed on groups of unspecified randomly chosen documents from a corpus of roughly one million government and military web pages, which seems to have been generated through recursive web crawling, and has likely changed since publication.

Our results also indicate that a two-pass approach, based on the execution of a "contextual" match only for the acronyms that were not discovered using a "canonical" match, may contribute to the improvement of the performance (increase recall, without drastic precision deterioration) of existing methods of acronym discovery, such as [5].

3.6 Future Work

A web-based version of the system, which accepts web documents available at user-entered URL-s is currently under development. This will create the possibility for user-initiated crawling of web documents, and the possibility to measure precision and recall based on actual user queries, a measure closer to the true usefulness of a system.

A complete search (rather than greedy) algorithm, using a scoring function for acronym expansion matches could be implemented, which may improve precision. A different syllable parser may be needed, as the speed of the one currently used would decrease dramatically the overall performance of the system, in conditions of an exhaustive search.

We would like to measure and possibly improve the robustness of our algorithm by testing it on larger corpora, as well as by applying it to other domains. We are trying to use our algorithms on the complete RFC corpus (containing complete documents, not just the abstracts) and on a large corpus from the medical domain.

The exploration of more linguistic properties of acronyms justifies more work, by executing an extensive analysis of the syntax of acronym definitions (e.g. looking for a noun phrase structure of acronym expansions), and learning more about the phonology of acronym expansions. We are currently trying to test and possibly enhance the assumptions of our algorithm against a large acronym definition database. Our initial difficulties here are related to performance issues of the syllable parser we are using and to the broad and – many times – subjective view of what is and what is not an acronym.

We would like to approach the problem of identifying equivalent expansions for given acronyms (e.g. "Multipurpose Internet Mail Extensions" or "Multimedia Internet Message Extensions" for the acronym "MIME"), and the problem of sense disambiguation when selecting the appropriate expansion of an acronym in a specific context (e.g. choosing "Digital Signature Algorithm" or "Directory System Agents" as an expansion for the acronym "DSA" in a given context) or domain.

The semantics of acronyms and their expansions, and their relationship to the semantic representations of domains that they belong to, seems a very promising area of investigation. The use of acronyms and their expansions as domain markers, or as aids in the classification of documents by domain, seems an intriguing problem that we would also like to approach.

3.7 References

[1] Acronym finder. http://www.AcronymFinder.com, June 2003.

[2] S. Bradner. Rfc 2026: The internet standards process–revision 3. The Internet Engineering Task Force; http://www.ietf.org, February 2004, 1997.

[3] Michael Hammond. Syllable parsing in english and french. Rutgers Optimality Archive, http://roa.rutgers.edu (February 2004), 1995.

[4] Michael Hammond. Parsing syllables: modeling ot computationally. Rutgers Optimality Archive, http://roa.rutgers.edu (February 2004), 1997.

[5] Leah S. Larkey, Paul Oglivie, M. Andrew Price, and Brenden Tamilio. Acrophile: An automated acronym extractor and server. In *Digital Libraries '00 - The Fifth ACM Conference on Digital Libraries (San Antonio, June 2-7, 2000)*, pages 205–214. ACM Press, 2000.

[6] Opaui guide to lists of acronyms, abbreviations, and initialisms. http://spin.com.mx/~smarin/acro.html, February 2004.

[7] J. Sykes, editor. *The Concise Oxford dictionary of current English.* Oxford University Press, 6 edition, 1976.

[8] Kazem Taghva and Jeff Gilbreth. Recognizing acronyms and their definitions. Technical Report 95-03, ISRI (Information Science Research Institute) UNLV, 1995.

[9] World wide web acronym and abbreviation server (wwwaas). http://www.ucc.ie/cgi-bin/acronym, June 2001.

[10] Stuart Yeates. Automatic extraction of acronyms from text. In *Proceedings of the Third New Zealand Computer Science Research Students Conference Hamilton, New Zealand, April 1999, University of Waikato*, pages 117–124, 1999.

[11] Stuart Yeates, David Bainbridge, and Ian H. Witten. Using compression to identify acronyms in text. Submitted to Data Compression Conference DCC 2000, Snowbird, Utah, 2000.

Chapter 4

Efficient Acronym-Expansion Matching for Automatic Acronym Acquisition

Reproduced with minor modifications from *Proceedings of the International Conference on Information and Knowledge Engineering, IKE 03, volume 1, pages 32–37. CSREA Press, 2003.* with permission.

ABSTRACT

Acronyms are a very dynamic area of many languages. An efficient dynamic programming algorithm for matching acronyms with their expansions by maximizing a linguistic plausibility score is presented and is found to be very accurate, to $F_{\beta=1}$=99.6% on a corpus of acronym definitions. Given its high precision, the algorithm can be used as a component in new or existing automatic acronym acquisition systems.

4.1 Definitions

Most *traditional* definitions of acronyms are restrictive, such as in [6], "word formed from the initials of other words", and do not account for a significant number of acronyms (XML, DNA, HTTP, to name just a few notorious examples). We submit that acronyms are a form of systematic abbreviation.

We define here (extensively) *acronyms* as abbreviations of complex, possibly multi-word expressions, through mechanisms based on the association of some letters in the acronym with some

of the words in the expression. We refer to the expression originating an acronym as the *expansion* of the acronym.

The system presented in this paper can only analyze acronyms for which letters of the acronym occur sequentially within the expansion (a subset of acronyms covered by the above definition).

We refer to a construction containing both an acronym and its expansion, with the purpose of introducing the meaning of the acronym as the *definition* of the acronym.

When showing the match between an acronym and its expansion, occurrences of matching letters of the acronym within the expansion are underlined.

The length of an expansion is denoted as M and the length of an acronym as n. Acronyms are denoted as an arrays $A[1..n]$ of letters, and expansions as an arrays $E[1..M]$ of letters.

4.2 Background

Acronyms can be a significant barrier for human users to understanding specialized text. [2], a comprehensive hardcopy acronym dictionary[1] for the English language, lists 420,000 entries, most of which are acronyms, and the most comprehensive online resource, Acronym Finder[2] lists over 330,000 entries.

The dynamic nature of this area of language, with many acronyms being created every year, creates a need for software tools that can help with the automatic acquisition of acronyms from large bodies of text. Coverage of other languages enforces this need, given the predominant English language focus of most existing acronym resources and the high cost of systematic manual acronym acquisition.

Performance figures from a number of reviewed automatic acronym acquisition systems are reduced (below) to a common denominator, by using the F_β measure[3].

[7] use a dynamic programming algorithm that only takes into account initials of words in the expansion. Their system achieves $F_{\beta=1} = 93.53\%$ on documents from a restricted domain of government studies, without counting misses on some difficult acronym-expansions pairs.

[9] and [10] present heuristic and statistical methods, respectively, with results that are not easily comparable or reproducible.

[1] This publication is currently at its thirty-second edition, with four editions in only the last three years.

[2] On the web at: http://acronymfinder.com (February 2004).

[3] $F_{\beta=1} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$

[5] use combinations between definition pattern-specific and context-specific approaches in a spectrum of 4 different algorithms and achieve a maximum $F_{\beta=1}$ = 88.43% on selections from a broad corpus of web-based military documents, without counting acronyms shorter than 3 letters and excluding from the evaluation distances longer than 20 words between an acronym and its expansion.

Chapter 3 postulates that acronym formation is a linguistic phenomenon, following hierarchical rules at the discourse, syntactic, lexical, morphological and phonological levels and, on this basis, achieves $F_{\beta=1}$ = 96.77% on text from a restricted domain of technical documents, using a 2-pass algorithm.

We submit that the main difficulties of automatic acronym processing are the correct acquisition of acronym definitions and the matching of acronyms with their expansions within definitions.

4.3 Methodology

In order to improve upon the performance and coverage of reviewed systems, we propose a modular two-tier approach (illustrated in figure 4.1) to the automatic acquisition of acronym expansions:

- identification of possible acronym definitions in text
- acronym-expansion matching

The identification of possible short distance acronym definitions, such as in [5] successfully relies upon regular expressions that describe short-distance acronym definition patterns (such as "Acronym (Expansion)"). Based on these results, we feel that this approach promises adequate performance, provided a robust acronym-expansion matching component is readily available.

Long distance acronym definitions, occurring far less frequently, account for at most 3% of the recall of the system presented in chapter 3. We acknowledge that more work needs to be done in this direction, but that is outside the scope of our current efforts.

Given a modular acronym acquisition system as defined above, the recall of the system will be directly related to the performance of the definition-matching module (provided acronym-expansion matches are extracted from their proposed definitions), and precision will be directly related to the performance of the acronym-expansion matching module (provided a high number of false definitions are not submitted).

We focus here on the acronym-expansion matching task, and try to identify an efficient and accurate solution.

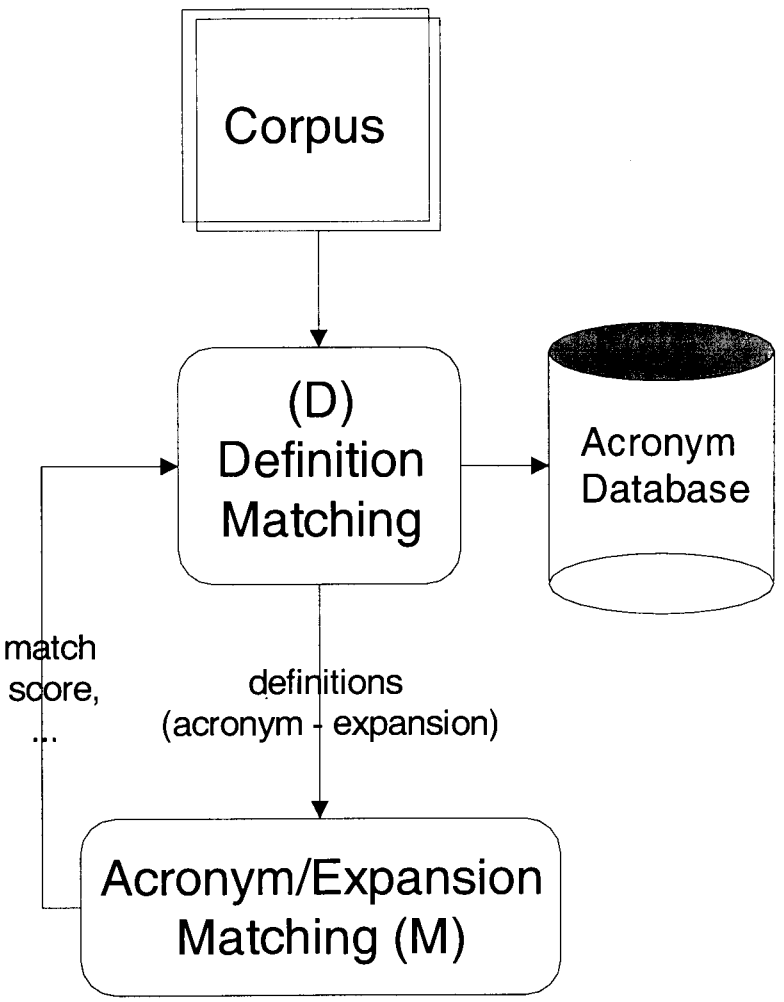


Figure 4.1: Modular approach to automatic acronym acquisition (Figure by author)

Acronym-expansion matching can be reformulated as a special case of the longest-common substring problem (presented and discussed in section A.7.2, with generalizations presented in section A.7.3), with the following added restrictions:

- all letters of the acronym occur in sequence (possibly discontiguous) within the expansion
- matches in the expansion must be "plausible" within the language of the acronym

To illustrate the second point, 'hypertext transfer protocol' is not a plausible match for HTTP in the English language, but 'hypertext transfer protocol' is. We submit that the condition of "plausibility" of an acronym-expansion match is a function of linguistic regularities in the language the acronym is defined in.

A brute force approach will enumerate all possible consecutive groups of letters in the expansion equal in length with the acronym and try to determine if they represent a match. This involves $\binom{M}{n}$ steps, yielding complexity $\Omega(M^3)$ for acronyms longer than two letters (prohibitively low performance).

A simple model that illustrates the worst case of the number of possible matches is the number of ways of matching the letters in a string S=$L_1 L_2 ... L_n$ (where L_i are n distinct letters) with letters from a string S^r=$L_1^r L_2^r ... L_n^r$, where L_i^r represents the letter L_i repeated r times. The number of possible matching scenarios of S within S^r is r^n.

For example, S="ABC" can match in 3^4=81 different ways the string S^4="AAAABBBBCCCC".

While more efficient algorithms (as the one shown below) can do better than the brute force approach, the potential for many alternative matches between an acronym and its expansion creates a need for ordering these using a plausibility or adequacy measure.

For naturally occurring acronym-expansion pairs, examples such as the 24 possible ways of matching 'CRIS' with "Concentric Research Information Service" are not uncommon (more than 5% of acronym definitions from our data analysis corpus have ambiguity degree at least 24). In such situations, the ambiguity degree provides an estimate of the computational complexity of the ordering of possible matches using a linguistic plausibility measure.

A dynamic programming approach that solves the acronym-expansion matching problem is presented, implemented and evaluated.

4.4 Acronym-Expansion Matching

Dynamic programming is an algorithmic paradigm that applies to so-called "optimization problems", with a solid mathematical foundation dating back to [1]. Dynamic programming has been used to solve a wide range of natural language processing problems, with well-known solutions such as the Viterbi and "forward" algorithms in speech recognition, machine translation and part-of-speech identification, the "minimum edit-distance algorithm" for spelling error corrections, and the CYK and Earley algorithms used in parsing (see [4] for overviews).

Formally, dynamic programming solutions can only be applied to problems that exhibit so-called "optimal substructure". In domains such as natural language processing, where empirical evidence is used to build theories of weak, interacting and many times contradictory violable constraints, the existence of optimal substructure needs to be postulated or can only arise within simplified models (such as the simplifying assumption of input symbols for the "forward" algorithm).

The assumption of "optimal substructure" stipulates that the most plausible match of an acronym-expansion pair implies the most plausible match for part of the acronym with a part of the expansion. This assumption is intuitive, but hard (if at all possible) to prove. We illustrate it by the following example:

If "hypertext transfer protocol" is the most plausible match for "HTTP", then "hypertext" is the most plausible match for "HT" and "transfer protocol" is the most plausible match for "TP".

We further submit that the level of success of our dynamic programming method of acronym-expansion matching provides empirical evidence for the predominant optimal substructure of acronym formation.

We treat acronym-expansion matching as a longest common substring problem, where letters in the acronym are matched with letters in the expansion, maximizing a "linguistic plausibility" score.

The linguistic plausibility score is built using a decreasing scale of matching letters in the acronym at:

- word boundaries (matches on word initials)
- noise word boundaries (matches on initials of conjunctions, prepositions or articles)
- morpheme boundaries (matches on first letters of intra-word morphemes)
- syllable boundaries (matches on first letters of intra-word syllables)
- matches on letters that do not fall into any of the above categories

We also include a "continuity" score, which favors contiguous matches within an expansion, such as the match of the final 'OM' in matching 'AVSCOM' with the expansion "aviation systems command". We calculate the continuity score as an average between the matching score of the first letter in the contiguous section and the current letter, weighted by an "adjacency factor" (1 is used).

A decreasing scale of linguistic plausibility of words in the expansion that are "skipped" in the match is also defined, as follows:

- Skip a punctuation-introduced word (such as skipping the word 'Pacific' in the match of 'AAMT' to the expansion "Asia-Pacific Association for Machine Translation"
- Skip a noise word (such as "for" in the previous example)
- Skip a regular (non-noise) word (such as skipping the words "Kwon Do" when matching 'ATA' to "American Tae Kwon Do Association")

Based on empirical observations within our development corpus, we define weights in a linear plausibility scale, which we tune using the following axioms:

1: Matching a letter adjacent to a word boundary-matching letter and skipping a noise word is less plausible than matching on a noise word boundary. Example: "Point of Contact" is a more plausible match for 'POC' than "Point of Contact".

2: Matching on a syllable/morpheme/word boundary is more plausible than matching on a letter/syllable/morpheme boundary and skipping a word.

3: Earlier matches are preferred.

Since the acronym definition-matching presented in [5] and 3 identifies both forward definition patterns (such as "Acronym (Expansion)", when the termination of an expansion is not obvious, but the beginning is) and backward definition patterns (such as "Expansion (Acronym)", where the end of an expansion is obvious, but the beginning is not), we design and implement two separate versions of our algorithm for forward and, respectively, backward matches. Our goal is to assign identical correct matches (on the correct letters in the expansion) using both the forward and the backward algorithms.

The implementation in the C++ language (including usage instructions) of our dynamic programming algorithm is available on our web site[4].

[4]On the web http://www.cs.sfu.ca/~manuelz/personal (February 2004).

We use the [3] syllabifier to generate syllable segmentation of words within expansions, manually built lists of noise words (conjunctions, prepositions, articles), and a manually-built list of English-language prefixes to suggest morphological decomposition.

The acronym–expansion matching algorithm builds a dynamic programming array, $D[n \times M]$, with values (excepting extremities) which are calculated as follows:

$$D(i,j) = \begin{cases} D_{i-1,j-1} + S(i,j) & \text{if } A[i] \text{ matches } E[j] \\ D_{i,j-1} & \text{otherwise} \end{cases}$$

where $S(i,j)$ is the matching score of $A[i]$ with $E[j]$. Word skipping penalties are also calculated at word boundaries, whenever applicable.

Figure 4.2 provides an illustration of a simple example of matching an acronym with its expansion. Word boundaries are indicated with 'w', morpheme boundaries with 'm', and syllable (phoneme) boundaries with 'p'.

In the matching array of the example, '*' indicates "no match", '\' indicates a match replacing the optimum, '+' indicates a match with lower score than the previous optimum, and '-' indicates the existence of an optimum match before the current position in the expansion. The matching positions within the expansion are indicated with letters of the acronym on the last line of the example.

```
  w  p m  p    w  p p p p p
  Multiprocess Communications
M \--------------++----------
P *****\--------------------
C ********\-----\------+------
  M    P       C
```

Figure 4.2: Acronym-expansion matching example (Figure by author)

4.5 Evaluation

We use a November 2001 version of the World-wide web acronym and abbreviation server corpus (WWWAAS) [8] containing a total of 17,529 entries, each entry being represented by an acronym-expansion pair.

We generate six separate batches of 100 randomly chosen entries each that we use as a development corpus. We use another 1,000 randomly chosen entries as an evaluation corpus.

Our evaluation corpus has no overlap with the development corpus. Entries in the evaluation corpus were never seen during development and testing of our software component.

Using information from the development corpus we develop a text segmentation utility that cleans up entries by removing extra explanations (often in square brackets) and entries that consistently fail to represent acronyms, such as coded standardized abbreviations (e.g. ISO country codes).

After running the segmentation utility on the evaluation corpus, we are left with 975 entries, from which we remove manually the ones that represent incorrect acronym-expansion pairs. This category includes typing mistakes in the corpus, such as 'DCCS' rather than 'DCSS' for "Discontiguous Shared Segments", or acronyms defined in other languages than the ones used in the entry, such as the English description "Cretaceous Tertiary Boundary" of the German acronym 'KTB'.

We also remove items that we do not consider to be acronyms, but coded abbreviations or standardized model names, such as T1 and X.400.

We are left with a total of 962 acronym-expansion pairs, to which we apply both the forward and backward matching versions of our acronym-expansion matching algorithm. All of the entries have English language expansions, with the exception of 81, which are defined in the French, German, Spanish and Serbian languages.

Many of the acronym-expansion pairs have additional text surrounding the expansion, which we do not eliminate at this stage.

644 of the entries have every letter of the acronym match the initial of a word in the expansion, consecutively, following the *traditional* definition of acronyms. The remaining 328 entries (approx. 34%) fall outside of it.

The system can not identify four of the acronyms as matching their corresponding expansions. In two of the situations, the algorithm did not take into account numeric representations of repetitive letters in acronyms, such as 'C4I' for "Command Control Communications Computers and Intelligence". Our algorithm also fails to account for symbolic abbreviations, such as 'X' for 'trans' in 'XPORT' for "TransPORT" and for unusual occurrences such as the doubling of royal plural initials in 'LLAARR' for "Leurs Altesses Royales".

Both our forward and backward matching algorithms fail for four entries. Only the forward algorithm fails for four other entries and only the backward algorithm fails for five other entries.

Three of the failures of both algorithms, two failures of only the forward algorithm and three failures of only the backward algorithm can be accounted for by incorrect text segmentation, and are corrected after proper segmentation of the entries.

		precision	recall	$F_{\beta=1}$
fully automatic segmentation	forward	99.18%	99.59%	99.54%
	backward	99.08%	99.59%	99.53%
after manual segmentation	forward	99.69%	99.59%	99.60%
	backward	99.69%	99.59%	99.60%

Table 4.1: Acronym-expansion matching performance on a corpus of acronym definitions (Table by author)

Two of the failures of the forward algorithm can be accounted for by incorrect morpheme segmentation of the expansion. In one of these, 'BPSG', was incorrectly matched to 'BoroPhosoSilicate Glass", since our system was not able to identify the morphemic components of the chemical compound word 'boro' + 'phoso[5]' + 'silicate'. The other failure was for an acronym in the German language, and was due to our lack of morphological resources for German, to help with the decomposition of compound nouns. Both of these failures are corrected when proper morphological segmentation of the entry was applied.

Two of the failures of the backward algorithm are caused by arguably ambiguous matching scenarios, such as 'INSTRAW' for "Institute for Research and training for the Advancement of Women" rather than "Institute for Research and training for the Advancement of Women" (forward and preferred matching).

We report the precision, recall and $F_{\beta=1}$ in Table 4.1 before and after the manual corrections for text segmentation, but without correcting for morphemic segmentation errors.

4.6 Conclusions

Our acronym-expansion matching algorithm has time complexity $\Theta(nM)$, uses $\Theta(nM)$ memory space and has high accuracy ($F_{\beta=1}$=99.6% for both forward and backward matching, after removing text segmentation errors).

The main challenge we have encountered is our lack of morphological resources with broad multilingual and multi-domain coverage, to help with the decomposition of morphological compounds. Similar difficulties arise as a result of a lack of phonological resources with broad multilingual coverage, even if our corpus contains only a small number of acronym definitions in languages other

[5]"phoso" is most likely an incorrect spelling of the morpheme "phospho."

than English, with no effect on the performance of our algorithm. Should these difficulties be surmounted, it is justified to expect near-perfect performance of acronym-expansion matching.

The presented system does not account for acronyms that use digits to indicate repetitive letters in the expansion (such as 'W3C' for the "World Wide Web Consortium"), or for symbolic abbreviations (such as 'C' for "see" or '2' for "to"). We believe that adding support for lists of symbolic expansions will accomplish this task.

Low efficiency of the syllabification component and of the morphological decomposition can be avoided by generating beforehand exhaustive lexicon-based lists of morphemic and syllable decompositions, which could be used during processing. Ad-hoc decomposition is only necessary for terms not covered within these lists.

We believe that the very high accuracy and good efficiency of our algorithm make it adequate for inclusion as a module in larger NLP systems, such as complete acronym acquisition and disambiguation frameworks. Our algorithm could also benefit existing systems, such as [5], by providing different quantitative thresholds for acronym-expansion matching within their canonical or contextual algorithms.

In Chapter chap:acro-acquisition, the acronym-expansion matching algorithm presented here is integrated with a definition matching module in a complete acronym acquisition system, using machine learning to identify regularities within acronym definitions in corpora.

4.7 References

[1] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[2] Mary Rose Bonk and Pamela Dear, editors. *Acronyms, initialisms, & abbreviations dictionary*. Gale Group, Detroit, 2001.

[3] Michael Hammond. Syllable parsing in english and french. Rutgers Optimality Archive, http://roa.rutgers.edu (February 2004), 1995.

[4] Daniel Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall series in artificial intelligence. Prentice Hall, 2000.

[5] Leah S. Larkey, Paul Oglivie, M. Andrew Price, and Brenden Tamilio. Acrophile: An automated acronym extractor and server. In *Digital Libraries '00 - The Fifth ACM Conference on Digital Libraries (San Antonio, June 2-7, 2000)*, pages 205–214. ACM Press, 2000.

[6] J. Sykes, editor. *The Concise Oxford dictionary of current English*. Oxford University Press, 6 edition, 1976.

[7] Kazem Taghva and Jeff Gilbreth. Recognizing acronyms and their definitions. Technical Report 95-03, ISRI (Information Science Research Institute) UNLV, 1995.

[8] World wide web acronym and abbreviation server (wwwaas). http://www.ucc.ie/cgi-bin/acronym, June 2001.

[9] Stuart Yeates. Automatic extraction of acronyms from text. In *Proceedings of the Third New Zealand Computer Science Research Students Conference Hamilton, New Zealand, April 1999, University of Waikato*, pages 117–124, 1999.

[10] Stuart Yeates, David Bainbridge, and Ian H. Witten. Using compression to identify acronyms in text. Submitted to Data Compression Conference DCC 2000, Snowbird, Utah, 2000.

Chapter 5

Automatic Acquisition of Long-Distance Acronym Definitions

Reproduced with minor modifications from *Proceedings of HIS 03 Hybrid Intelligent Systems, Melbourne 2003, pages 582–592. IOS Press, 2003.* with permission.

ABSTRACT

Acronyms are a very dynamic area of the lexicon of many languages. A hybrid, modular methodology for the acquisition of acronyms is presented, which uses an existing acronym-expansion matching component, and machine learning in two separate phases for the identification of long-distance acronym definition patterns.

The resulting system, using Support Vector Machines (SVM) is trained on 600 news stories from the Wall Street Journal component of the Penn Treebank corpus using a number of lexical, syntactic, and acronym-expansion matching features. Statistical cooccurrence information for acronym-expansion pairs is extracted from search engine "hit counts".

The system achieves $F_{\beta=1}$=92.38% on 400 news stories from the same source and has good asymptotic efficiency, making it adequate for the automatic extraction of acronyms even from noisy sources, such as newspaper text.

5.1 Background

Acronyms are defined here, extensively[1], as complex abbreviations of expressions (possible multi-word), through mechanisms of association of letters in the acronym with words in the expression. The expression originating an acronym is refered as the *expansion* of the acronym.

Constructions containing both an acronym and its expansion, with the purpose of introducing the meaning of the acronym are refered as *definitions* of acronyms. When showing the match between an acronym and its expansion, occurrences of matching letters of the acronym within the expansion are underlined.

Acronyms are arguably the most dynamic area of the lexicon of numerous languages. For English, the most comprehensive acronym dictionary[2], [1] contains "more than 450,000 acronyms, abbreviations and initialisms", most of them acronyms, whereas the most comprehensive online resource, Acronym Finder[3], claims coverage of over 275,000 acronyms. New acronyms are constantly being created, which justifies a need for software tools that can assist with the automatic acquisition of acronyms from large text collections. Coverage of other languages enforces this need, given the predominant English language focus of most existing acronym resources and the high cost of systematic manual acronym acquisition.

Acronyms present two distinct problem areas: *Acronym Acquisition*, where acronyms are collected and matched with their expansions in text, and *Acronym Disambiguation*, where the appropriate sense (expansion) for an acronym occurrence is selected.

Acronym disambiguation, which is mentioned but not approached here, has been studied in the medical domain by [7], with good level of success (89% accuracy) but remains an open problem for general text.

Current acronym acquisition systems use exclusively either classical algorithms such as dynamic programming, as in [9], chapter 4, pattern matching heuristics (sometimes with an AI flavor), as in [11], [5], 3, or statistical methods, as in [12]. A more in-depth overview of reviewed systems is presented in Section 1.1.

A modular approach to acronym acquistion, is presented here, as illustrated in Figure 4.1. The data flow of the experimental setup used to evaluate this system is presented in Figure 5.1.

[1]Classic definitions, such as in [8], of an *acronym* as a "word formed from the initials of other words", do not account for a significant number of acronyms (XML, DNA, HTTP, to name just a few notorious examples).

[2]This publication is at its thirty-second edition, with four editions in just the past three years.

[3]On the web http://acronymfinder.com (February 2004).

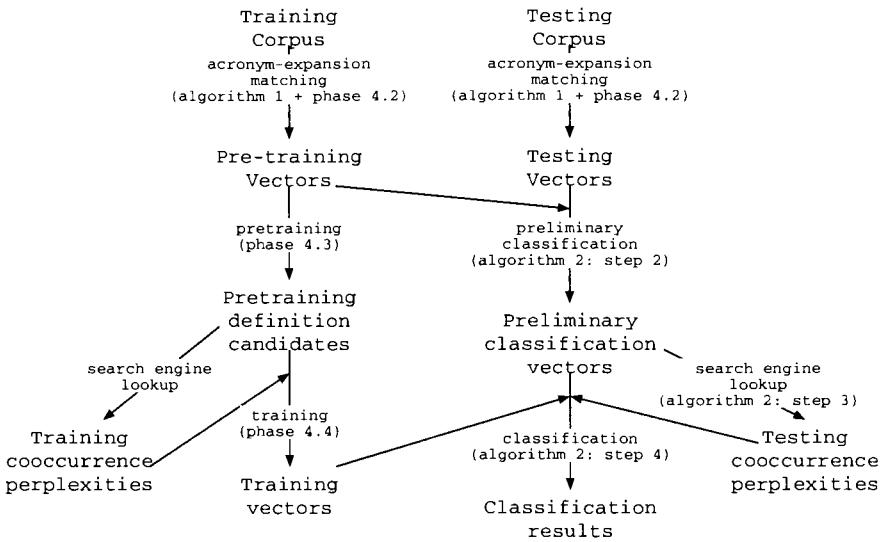


Figure 5.1: Evaluation phases and data flow for acronym acquisition (Figure by author)

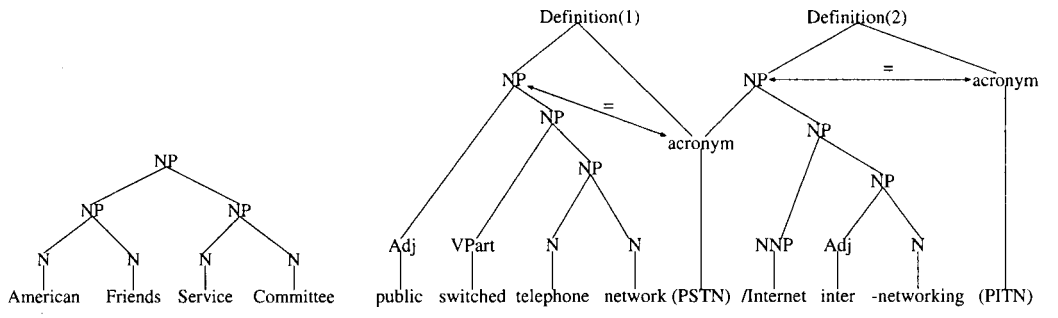
Acronym acquisition by humans relies on the recursive nature of natural language, on learning statistical regularities, and on phenomena that can be modeled through hard (symbolic) computing algorithms. The hybrid nature of this problem yields itself to a hybrid methodology, which is proposed here as a combination of: *AI techniques* for parsing of text into constituents and the identification of patterns of possible short-distance acronym definitions, *hard computing techniques* (dynamic programming) for efficiently establishing the adequacy of acronym-expansion matches, *machine learning* of long-distance acronym definition patterns, and *statistical methods* for the validation of results.

5.2 Assumptions about the nature of acronym definitions

The following underlying assumptions are proposed about the nature of acronym definitions.

Hypothesis 5.1 *Matching acronyms with their expansions can be modeled efficiently and with very high accuracy through dynamic programming algorithms.*

The algorithm presented in 4 is used as a component of the system.



(a) Acronym expansion.

(b) Cascading acronym definitions.

Figure 5.2: Examples of phrase structure of acronym expansion and definitions (Figure by author)

Hypothesis 5.2 *Acronym expansions have underlying syntactic phrase structure, and generally do not cross the boundaries of other phrasal constituents.*

An example (for 'AFSC', an acronym for "American Friends Service Committee") is provided in Fig. 5.2(a). Counterexamples, such as the cascading acronym definitions[4]: "Public Switched Telephone Network (PSTN)/Internet Inter-Networking (PINT) Working Group", with constituent structure illustrated in Fig. 5.2(b) are very unusual.

Hypothesis 5.3 *There are relatively few, well described short-distance acronym definition patterns, which can be successfully modeled by regular expressions, [5] and chapter 3.*

Hypothesis 5.4 *Long-distance acronym definitions are a form of discourse cohesion*[5].

The following example[6] illustrates the definition, through anaphora, of 'TFTP', an acronym for "trivial file transfer protocol". Coreferential entities are marked with ⌊anaphor $_{id}$⌋ and ⌈referent $_{id}$⌉.

> The ⌈$_1$ Trivial File Transfer Protocol $_1$⌉ [1] is a simple, lock-step, file transfer protocol which allows a client to get or put a file onto a remote host. One of ⌊$_1$ its $_1$⌋ primary

[4]Extract from "RFC2458: Toward the PSTN/Internet Inter-Networking –Pre-PINT Implementations" on the web http://ietf.org (February 2004).

[5][3] is suggested as a systematic review of discourse cohesion phenomena from a linguistic point of view.

[6]Extract from "RFC1783: TFTP Blocksize Option"; on the web http://ietf.org (February 2004).

uses is the booting of diskless nodes on a Local Area Network. ⌈2 ⌊1 TFTP 1⌋ 2⌉ is used because ⌊2 it 2⌋ is very simple to implement in a small node's limited ROM space.

The second reference to entity 1 is made through the acronym 'TFTP', which also indicates the introduction, in a hypothetical discourse knowledge repository of the acronym definition: 'TFTP' = "trivial file transfer protocol."

I submit here that acronym definitions are, in general, either a form of *anaphora* (when the expansion precedes the acronym) or *cataphora* (when the acronym precedes the expansion). Further, in cases when the meaning of an acronym is assumed to be well known to the reader, the relationship between the acronym and its expansion, when present in the same text, is one of *coreference*. For example, it can be thought that the meaning (expansion) of the acronym 'U.S.' is well known to the readership of the *Wall Street Journal*. Cooccurrences in text of 'U.S.' and its expansion "United States" indicate coreference, justified by a reduction of repetitiveness, rather than the need to introduce to the reader the definition of the acronym 'U.S.'

Hypothesis 5.5 *General world knowledge is necessary for judging the likelihood that a certain acronym-phrase pair represents an acronym definition.*

In the following hypothetical example, most readers will arguably not assume that 'CIA' is an acronym for "Contrary to international allegations", even if the initials match.

Contrary to international allegations, the CIA ... (5.1)

5.3 Methodology

A system using machine learning is proposed, that can be trained on parsed text containing labeled acronym-expansion pairs representing acronym definitions. The main goal is for the system to learn to automatically identify long-distance acronym definitions in previously unseen text.

During the development of the system a *development corpus* is used, containing 400 parsed, randomly selected Wall Street Journal news stories from the Penn Treebank corpus, [6]. All occurrences of acronym definitions are manually marked as acronym-expansion pairs and differentiated from cases where an acronym and its expansion are only coreferential expressions.

The system processes text in the phases described below and shown in Figure 5.1.

5.3.1 Generation of acronym definition candidates from text

Algorithm 5.1 (Acronym definition candidates) *:*

1. *Identify all acronym candidates from the text by matching capitalization patterns*[7].
2. *List all syntactic constituents of the text (words, phrases, sentences).*
3. *For each pair: acronym candidate-syntactic constituent, attempt a match using the acronym-expansion matching algorithm described in chapter 4.*
4. *Keep only pairs that result in strictly positive matching scores, i.e. where letters in the expansion candidate match all letters in the acronym candidate. The result is the list of* acronym definition candidates.

Algorithm 5.1 results in $O(a \cdot T \cdot logT)$ acronym definition candidates and has complexity: $O(l \cdot \overline{S} \cdot T \cdot logT)$, where l is the sum of the lengths of all acronyms in the text, $\overline{S}$ is the average sentence length, T is the size of the text in tokens, and a is the number of acronyms.

5.3.2 Calculation of feature values

For each resulting acronym definition candidate, the values of 46 of features are calculated, in constant time for each definition candidate:

- *Acronym and expansion candidate features*, such as length, number of noise words.
- *Acronym-expansion matching values*, such as the matching score and number of words matched in the expansion candidate.
- *Phrase structure tree features*, such as the distance in words, phrases and sentences between the acronym and the expansion candidates.
- *Lexical features*, such as the part-of-speech values for the acronym and the first element in the expansion candidate.
- *Contextual features*, such as the similarity between the immediate contexts of the acronym and the expansion candidates, also whether the definition candidate can be modeled by one of the regular expressions describing short-distance acronym definitions, in Assumption 5.3.

[7]Capitalization patterns alone are not enough to differentiate between an acronym and words capitalized for emphasis.

5.3.3 Pretraining

The system uses a machine learning approach based on pattern recognition using Support Vector Machines (SVM), a statistical learning technique based on the work of Vapnik [10]. SVM are used in pattern recognition problems, [2] The system described here uses the SVMlight[8] implementation, [4].

The pretraining phase has as objective the reduction of the search space of acronym definitions, without a dramatic reduction in recall. In other words, to learn to remove all unlikely acronym definition candidates, removing as few genuine acronym definitions as possible.

A *training corpus* is used during the pretraining phase, containing syntactically parsed text, in which all acronym definitions and acronym-expansion coreference cases have been manually marked. All acronym definitions are used as positive examples, and all definition candidates which are not definitions (*false definitions*) as negative examples for pretraining. All cases of acronym-expansion coreference are ignored.

It can be expected that the number of false definitions is significantly larger than the number of definitions. Indeed, for the development corpus, out of 94,404 definition candidates, just 105 are genuine acronym definitions, and an additional 43 are cases of acronym-expansion coreference.

This discrepancy creates a need to increase the comparative weight of positive examples (the *positive emphasis*) in training[9]. During development, values for the positive emphasis between 1 (no emphasis) and 1,000 (roughly equal to the proportion of negative vs. positive examples) have been used. On the development corpus a positive emphasis of 400 was found to be sufficient for capturing most of the correct examples (consistently over 95%), yet was low enough to reduce significantly the definition candidate search space (to less than ten times the number of genuine definitions).

The same data used for pretraining is also evaluated against the *pretraining vectors* (SVM parameters following pretraining). All definition candidates that classified as positive using the pretraining vectors are considered *pretraining definition candidates*, a reduced set – compared to the original set of definition candidates – of all definition candidates with positive *pretraining classification scores*.

[8] On the web http://svmlight.joachims.org/ (February 2004).

[9] In the development corpus, if the weight of positive and negative examples is equal, the system ends up invariably classifying all occurrences as negative, as is common in sparse data problems.

5.3.4 Training using cooccurrence information

Following Assumption 5.5 above, information in a given text alone is not enough to identify all acronym definitions in that text. General world knowledge, on the other hand, is very difficult to obtain with enough coverage and accuracy. Testing on the development corpus of the performance of classification following pretraining only, with different positive emphasis, consistently yields performance ($F_{\beta=1}$) in the mid-70%'s, which reinforces this assumption.

A method for bootstrapping the acquisition of world knowledge about acronyms and their expansions is proposed, by measuring the expectation of cooccurrence of the acronym candidate with the expansion candidate and comparing it with the observed cooccurrence of the acronym and expansion candidates.

A given domain of documents of size D is considered, where an acronym candidate A occurs within N_A documents, an expansion candidate E occurs within N_E documents, and N_{AE} documents contain occurrences of both A and E. The probability that a given document contains an occurrence of A is: $P_A = \frac{N_A}{D}$ and the probablity that a given document contains E is: $P_E = \frac{N_E}{D}$.

Supposing that A and E are conditionally independent, the probability that a given document contains both A and E is: $P_{AE} = P_A \cdot P_E = \frac{N_A \cdot N_E}{D^2}$ and the expectation, $\rho(A, E)$ of the number of documents containing both A and E is: $\rho(A, E) = P_{AE} \cdot D = \frac{N_A \cdot N_E}{D}$.

The normalized[10] *perplexity*, $\pi(A, E)$ of encountering the acronym and expansion candidate together, is calculated as the logarithm of the observed number of cooccurrences over the expected number of conditionally independent cooccurrences of the acronym and expansion candidates:

$$\pi(A, E) = log_{10}\left(\frac{N_{AE}}{\rho(A, E) + 1} + 1\right) = \log_{10}\left(\frac{N_{AE}}{\frac{N_A \cdot N_E}{D} + 1} + 1\right) \tag{5.2}$$

Since information about a specific domain needs to be collected from large text repositories, which are prohibitively expensive to build, the system uses *number of hits* information that can be obtained from search engines. At the time of writing this article, the Google™ search engine[11] offers a beta version of a web API, which allows – with some restrictions – the execution of bulk search queries. This component is integrated into the system.

To exemplify, the perplexity of encountering the acronym 'SARS' and its expansion "Severe Acute Respiratory Syndrome" can be calculated as $\pi(\text{'SARS'}) \approx 2.7279$, by using the number of

[10] "+1" is used as a smoothing factor, for removing division by zero, and respectively, logarithm of zero.

[11] On the web http://www.google.com (February 2004). All supporting searches were executed during June and July 2003.

hits and total number of documents ($D_{\text{Google}} \approx 3 \cdot 10^{12}$) indexed by the Google search engine. In other words, the cooccurrence of the acronym 'SARS' and its expansion is more than 500 (or $> 10^{\pi(\text{'SARS'})}$) times more likely than chance, indicating strong correlation between the two phrases.

The system is trained a second time on all the pretraining definition candidates (obtained in phase 5.3.3), using as features their SVM pretraining classification scores, and the cooccurrence perplexity, $\pi(A, E)$, calculated as above. The positive emphasis parameter of the training is set to an integer rounded up from the number of *pretraining definition candidates in the training corpus* divided by the number of acronym-expansion definitions in the training corpus. The result of the training phase is a set of *classification vectors*.

5.3.5 Classification

Acronym definitions candidates from previously unseen evaluation text are classified using the following algorithm:

Algorithm 5.2 (Discovery of acronym definitions) *:*

1. *Identify all definition candidates with strictly positive acronym-expansion matching scores.*
2. *Classify all definition candidates (from step 1) using the pretraining vectors obtained in phase 5.3.3. Keep only positive matches, and their* preliminary classification scores.
3. *For each pair of acronym-expansion candidates (A,E) obtained in step 2, calculate* $\pi(A, E)$.
4. *Classify all acronym-expansion pairs obtained in step 2, using the classification vectors obtained in phase 5.3.4, using features* $\pi(A, E)$ *and the preliminary classification scores calculated in step 2.*

The complexity of the Algorithm 5.2 above is the same as for the Algorithm 5.1.

An additional $O(a)$ search engine lookups is executed in step 4, since each acronym definition yields at most one acronym, and the number of definitions candidates obtained in step 2 of the algorithm is bound by a constant times the number of genuine definitions (during evaluation of the system, the value of this constant is 3).

Table 5.1: Results of evaluation of acronym acquisition. Performance reported for **bold** columns (Table by author)

following pretraining and preliminary classification	positive emphasis	2	4	5	50	200	**400**	800
	recall	50.50	73.73	76.76	91.83	96.96	**97.97**	100.00
	precision	87.71	80.21	76.76	48.38	36.50	**32.88**	29.37
	$F_{\beta=1}$	64.10	76.83	76.76	63.37	53.04	**49.24**	45.40
following training and classification	positive emphasis	1	2	**3**	4	10	20	30
	recall	82.82	88.88	**91.91**	91.91	94.94	97.97	97.97
	precision	92.13	92.63	**92.85**	88.34	77.04	68.30	65.98
	$F_{\beta=1}$	87.23	90.72	**92.38**	90.09	85.06	80.49	78.85

5.4 Evaluation

The system is trained and tested on text of Wall Street Journal news stories from the Penn Treebank corpus, [6].

The *training corpus* contains all 400 stories from the development corpus, and an additional randomly chosen 200 stories. The system is evaluated using an *evaluation corpus*, with 400 additional randomly chosen stories. The evaluation corpus was never seen during development.

The training and evaluation corpora are manually marked, to identify acronym-expansion pairs that represent definitions. Cases of cooccurrence which are not definitions, where the acronym and expansion are in a relation of coreference are also marked. Corefering acronym-expansion pairs are removed from the evaluation.

The performance of the pretraining phase is calculated and plotted (for positive emphasis between 2 and 800, on a logarithmic scale) in figure 5.4, by evaluating the performance of the classification of all acronym-expansion pairs using only pretraining vectors. Pretraining achieves precision=32.88%, recall=97.97% and $F_{\beta=1}$=49.24% for a positive emphasis of 400 (the value used in the complete evaluation of the system). The highest performance achieved by the pretraining phase alone (comparing $F_{\beta=1}$) is for a positive emphasis of 4: precision=80.21%, recall=73.73%, $F_{\beta=1}$=76.83%.

The performance of the complete system (including training) is evaluated for a training positive emphasis of 3 (calculated after pretraining). Results: precision=92.85%, recall=91.91% and $F_{\beta=1}$=92.38%. Values are also calculated using positive emphasis between 1 and 10, 20 and 30. Results are plotted in figure 5.4.

Some partial results following preliminary classification and complete classification on the testing corpus are represented in Table 5.1.

Two definitions are excluded during pretraining:

- 'DNA' for "deoxyribonucleic acid", is likely excluded ("justifications" of decisions by support vector machine classifiers are not available) due to the low preference given to matches on syllables (de-oxy-ri-bo-nuc-le-ic).
- 'TNT' for "Turner Network Television" is likely excluded because of its embedding into a much bigger expression (Turner Broadcasting 's Turner Network Television channel), represented as a single Penn Treebank node.

Six more definitions are not found during the classification phase: three occurrences of 'CD', for "certificate(s) of deposit"; 'AM' for "Mayer Amshel"; 'NP' for "National Pizza", and 'AT&T' for "American Telegraph Telephone".

All these definitions are likely discarded because of relatively low values (close to 1) for the perplexity $\pi(A, E)$. In the case of the last definition, the phrase "American Telegraph *and* Telephone" is the correct expansion for AT&T, and occurs 392 times within documents indexed by Google, compared to just 6 times for the incorrect one, encountered in the evaluation corpus.

The following acronym-expansion pairs are incorrectly classified as definitions, in most cases because of a high pretraining classification score: 'USS' for "US and Soviet"; 'LTCB' for "Ltd Chicago branch"; 'US' for "Union Bank of Switzerland"; 'S&P' for "Security Pacific"; 'NJ' for "New York until January"; 'CBS' for "Chairman Burt Sugarman", and 'MGM' for "Mr Guber and Mr".

It is expected that inclusion of additional statistical features (future work) in the training and classification phases will discard some of the incorrect entries. During development, additional statistical features have been added, leading to an increase in precision at the expense of the recall, attributed to the relatively small size of the training corpus.

I expect that the inclusion of additional statistical features will only be effective if combined with an increased size of the training corpus.

5.5 Conclusions

The system presented achieves good performance on data that is arguably very noisy. Out of the 291 acronym definitions included in the training and evaluation corpora combined, for only 41 of

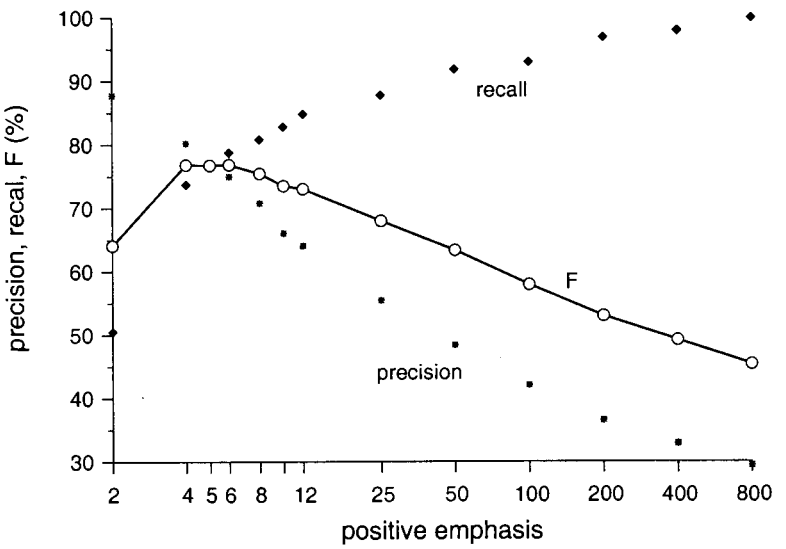


Figure 5.3: Acronym acquisition precision, recall and $F_{\beta=1}$, after pretraining, as functions of positive emphasis (Figure by author)

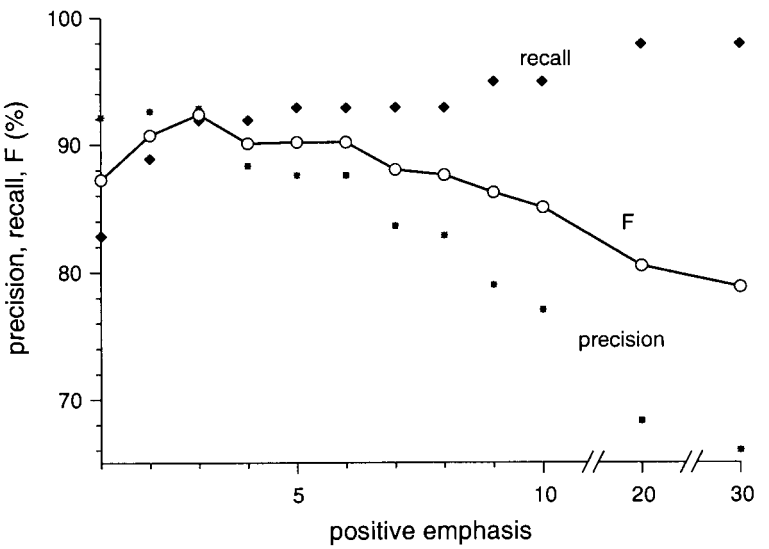


Figure 5.4: Acronym acquisition precision, recall and $F_{\beta=1}$, after training, as functions of positive emphasis (Figure by author)

them the acronym occurs within a distance of four tokens or less from the expansion. These are the only ones that qualify as short-distance acronym definitions. 201 acronyms (or almost 70%) occur within 20 tokens or more from their expansion, a limit over which misses are not counted in the evaluation of the system of [5], which reports lower performance (maximum $F_{\beta=1}$ = 88.43%) on different, arguably clearer, data.

Performance is also very high compared with state-of-the-art anaphora or coreference resolution systems, mainly because of the availability of hard computing criteria for judging the adequacy of acronym-expansion matches (unlike less specific matching restrictions for pronominal anaphora), and the ability to statistically quantify correlations between acronym and expansion occurrences.

While most acronyms within the evaluation corpus match initial letters of words in the expansion (falling within the classical definition of acronym), occurrences such as 'UAL' for "United Airlines" are successfully found by the system.

I suggest that training on a larger corpus, possibly within an interactive, human-assisted environment will lead to higher performance of the system. The release of a web-crawler-based version of the system, currently under development, will help achieve this.

The system is evaluated on "golden standard" syntactically parsed data. Processing raw text and incorporating a parser (future work) are expected to decrease performance. I suggest that this decrease will not be significant, or even present, since good performance partial parsing – currently achievable – is the main requirement for the generation of good quality syntactic features.

The absence of high-performance morphological and syllabification resources, inherent in the acronym-expansion matching component presented in chapter 4, is critical for domains or languages with rich compound or agglutinative morphology.

The main bottleneck of the system is the Google web API, currently only allowing 1,000 queries per day, and processing queries through a web interface at a speed of 1-4 per second. Building, or gaining unrestricted broadband access to search-engine-like document indexes, at least for specialized domains of usage will remove this difficulty. Such a scenario will also remove the need for a preliminary pretraining phase, allowing for the inclusion of cooccurrence perplexity information with all acronym-expansion candidates.

In order to prove the robustness of the results, the system needs to be evaluated on text from other domains or languages. Such evaluations for text in the German language and for medical text are currently being undertaken.

5.6 References

[1] Mary Rose Bonk and Pamela Dear, editors. *Acronyms, initialisms, & abbreviations dictionary.* Gale Group, Detroit, 2001.

[2] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[3] Michael A.K. Halliday and Ruqaiya Hasan. *Cohesion in English.* Number 9 in English Language Series. Longman, 1976.

[4] Thorsten Joachims. Making large-scale svm learning practical. In B. Schölkopf, Christopher J.C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.

[5] Leah S. Larkey, Paul Oglivie, M. Andrew Price, and Brenden Tamilio. Acrophile: An automated acronym extractor and server. In *Digital Libraries '00 - The Fifth ACM Conference on Digital Libraries (San Antonio, June 2-7, 2000)*, pages 205–214. ACM Press, 2000.

[6] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[7] Serguei Pakhomov. Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL) Philadelphia, July 2002*, pages 160–167, 2002.

[8] J. Sykes, editor. *The Concise Oxford dictionary of current English.* Oxford University Press, 6 edition, 1976.

[9] Kazem Taghva and Jeff Gilbreth. Recognizing acronyms and their definitions. Technical Report 95-03, ISRI (Information Science Research Institute) UNLV, 1995.

[10] Vladimir Vapnik. *Statistical Learning Theory.* Wiley, New York, Chichester, 1998.

[11] Stuart Yeates. Automatic extraction of acronyms from text. In *Proceedings of the Third New Zealand Computer Science Research Students Conference Hamilton, New Zealand, April 1999, University of Waikato*, pages 117–124, 1999.

[12] Stuart Yeates, David Bainbridge, and Ian H. Witten. Using compression to identify acronyms in text. Submitted to Data Compression Conference DCC 2000, Snowbird, Utah, 2000.

Chapter 6

Multilingual Acronym Regularities

In this chapter, difficulties of acronym acquisition in different languages are discussed and quantified in terms of the following measures: optimal matching score, accidental match probability and matching ambiguity.

A corpus-based study of acronyms in the English language is conducted and its results indicate that densities of accidental match probability and matching ambiguity are consistent with Zipf-Mandelbrot distributions, common in sparse data problems.

A comparative corpus-based study of acronym definitions in eleven languages is conducted, using a multilingual list of acronym definitions and the results support the validity of the universal theory of acronyms.

A dynamic programming acronym-expansion matching algorithm (ACE) using slightly different ordering and weights for different languages successfully matches $F_{\beta=1}$=98.58% acronym-expansion pairs in Russian and with $F_{\beta=1}$ over 99% for Spanish, Danish, German, English, French, Italian, Dutch, Portuguese, Finnish, and Swedish.

This work indicates the applicability of acronym acquisition methods based on the universal theory of acronyms to languages other than English.

6.1 Introduction

I have proposed in Chapter 2 a *universal theory of acronyms*, which attempts to explain all acronyms in all languages. The theory is based on a number of hypotheses, which support a system of violable rules. Since the theory was developed based on observations of specific cases, for it to be credible, its validity must be tested on actual data.

A system providing acronym-expansion matching in English, presented in chapter 4 uses a subset of the rules and achieves excellent results ($F_{\beta=1}$=99.6%) matching acronym-expansion pairs from a corpus of acronym definitions [145]. The same technology is used as a component in a hybrid system with a machine learning component, presented in chapter 5, achieving performance of $F_{\beta=1}$=92.38%. These results provide encouraging validation for the adequacy of the theory of acronyms for English.

Since all previous reviewed research on acronyms is exclusively focused on the English language, evaluations on other languages is necessary. Comparative quantitative analyses of acronyms formation regularities will also provide an understanding of whether acronym acquisition systems built for the English language are expected to perform adequately in other languages.

6.2 Acronym Acquisition Difficulties

Intuitively, acronym acquisition is a relatively simple problem for acronyms which match only initials (*matching initials*) and are declared in the immediate vicinity of their expansion, through definitions which can be modeled through regular expressions, [82] and in chapter 3.

Symbolic matching (rule 2.10) is very infrequent in the English language, as shown in chapter 4 and modeling its most representative cases is no more difficult than building comprehensive lists of symbolic matching expressions, by expanding upon work in areas such as creative spellings [102] and numeric multiplicative matching [82].

A number of more difficult problems arise, however, for:

- Matches of acronym letters on letters other than initials of words in the expansion.
- Long-distance acronym definitions, very common in some domains, such as newspaper text. This problem is solved with a reasonable level of success in chapter 5.
- Acronym inversion (rule 2.14) occurs either accidentally (hypothesis 2.6) or, in some languages, systematically (hypotheses 2.4 and 2.5).

Metrics which quantify these difficulties are introduced next, and solutions to overcome them are proposed.

6.2.1 Ambiguity

As discussed earlier, matching acronym letters to expansion letters other than word initials (*matching internal letters*) can be due to any or more of:

- Preference for acronyms longer than two letters (hypothesis 2.1).
- Preference for increased pronounciability (hypothesis 2.2).
- Conflict with already established acronym forms, in the target domain (hypothesis 2.3).

Matching internal letters is computationally more difficult than matching initials, due to:

- Increase in the computational complexity of the brute force approach for matching acronym-expansion pairs, which grows with the length of the acronym (n) and is proven to involve $\binom{M}{n}$ steps, which, for acronyms longer than two letters is $\Omega(M^3)$ in the size (M) of the expansion, as discussed in chapter 5.
- Increase in the likelihood of spurious matches between the acronym and unrelated expressions in the target text. For relatively short segments of newspaper text from the Penn Treebank [91], the number of expressions that represent matches for acronyms encountered in the text is 99,404 compared to only 105 genuine definitions, as shown in chapter 5.

The computational complexity of the brute force approach (enumeration of all possible matches) can be modeled by a quantitative measure of *ambiguity degree*.

Definition 6.1 (Ambiguity degree) *The* ambiguity degree *of a given acronym-expansion candidate pair is the the number of ways in which the letters in the acronym can match letters in the expansion.*

For example, the acronym 'HTTP' can be matched to its expansion in three ways, only one (6.2) correct:

'HTTP' for "hypertext transfer protocol" (6.1)

'HTTP' for "hypertext transfer protocol" (6.2)

'HTTP' for "hypertext transfer protocol" (6.3)

A dynamic programming algorithm is proposed, wich calculates the ambiguity degree for the acronym-expansion pair, by successively filling values in an $acronym \times expansion$ array. The

value of element $B_{i,j}$ in the array represents the ambiguity degree of the pair containing the first i letters of the acronym: $A[1..i]$ matched to the first j letters (excluding spaces) of the expansion: $E[1..j]$, calculated as follows (excluding extremity conditions): $B_{i,j} = B_{i-1,j} + X(i,j)$, where:

$$X(i,j) = \begin{cases} B_{i-1,j-1} + 1 & \text{if } A[i] = E[j] \\ 0 & \text{otherwise} \end{cases} \tag{6.4}$$

Given the hypothesis of existence of a set of universal explanatory rules for acronyms, with possible language specific activations and orderings (hypothesis 2.7), the minimum set of broken rules for a given acronym-expansion pair can be calculated, in absence of other information, only after the enumeration of all possible matches, given by the ambiguity degree.

6.2.2 Optimal Substructure

I suggested in chapter 4 avoiding the possible resulting computational explosion through the association of *weights* with rules, which simulate a *linguistic plausibility score* and the use of dynamic programming algorithms. Such efficient dynamic programming solutions to the acronym-expansion matching problem are possible only if a hypothesis of optimal substructure is valid.

Hypothesis 6.1 (Acronym-expansion matching optimal substructure) *The linguistic plausibility score associated with ordering and weights of the set of universal acronym formation rules exhibits* optimal substructure.

Dynamic programming or greedy algorithms (such as presented in chapters 3 and respectively 4 have as prerequisite the existence of optimal substructure, which can be proven mathematically for classical problems such as matrix chain product optimization, the construction of optimal binary search trees, or optimal binary codes (reviews are available [28]).

For natural language problems, however, optimal substructure cannot be proven and can only be postulated. Its validity can be infered only through success in modeling computationally the sought after phenomena. I submit that the excellent performance of the dynamic programming algorithm for acronym-expansion matching in English 4 supports hypothesis 6.1 (optimal substructure) for acronyms in the English language.

The validity of this hypothesis in other languages is investigated further, as a prerequisite to efficient acronym acquisition systems in those languages.

6.2.3 Accidental Matches

Any given acronym matches an expression drawn at random from a given corpus, with a probability that is non-zero after excluding trivial situations (such as the expressions shorter than the acronym).

Definition 6.2 (Accidental matching probability) *The* accidental matching probability *associated with an acronym-expression pair is the probability that an expression of a given size and structure (length of constituent words), randomly drawn from a given corpus, matches a given acronym form.*

The accidental match probability can be high for short (e.g. two letter) acronyms, and also for longer acronyms formed from letters that are frequent in the target language, and longer possible expansions.

A dynamic programming algorithm is proposed, which calculates an approximation to the accidental match probability for an acronym-expansion pair, as the probability that an expression, generated using letter occurrence probabilities for the given corpus which fits the word number and lengths for the given expansion, matches the given acronym.

The algorithm successively fills values in an $acronym \times expansion$ array. The value of element $P_{i,j}$ in the array represents the accidental match probability for the pair containing the first i letters of the acronym: $A[1..i]$ and the first j letters (excluding spaces) of the expansion: $E[1..j]$. $P_{i,j}$ is calculated as follows (excluding extremity conditions):

$$P_{i,j} = P_{i-1,j} \cdot [1 - Y(i,j)] + P_{i-1,j-1} \cdot Y(i-1,j) \tag{6.5}$$

In equation (6.5) $Y(i,j)$ is the probability that letter $A[i]$ of the acronym occurs at position j of random generated text, with letter occurrence probabilities calculated for the domain of the acronym-expansion occurrence. Different letter occurrence probabilities are allowed at word, morpheme, syllable boundaries or without restrictions.

The accidental match probability calculated this way is an approximate measure of the probability that the acronym matches random text drawn from the domain (supposedly by an over-zealous definition-matching component). This will help a modular acronym acquisition system (shown in figure 4) fine-tune how aggressively it selects acronym definitions, based on the probability of accidental acronym-expansion matches.

For example, if an acronym definition matching module outputs a possible acronym-expansion pair: 'AEI' for "asynchronous transfer mode electrical interface" with low confidence, this acronym-expansion can be discarded, since the accidental match probability is very high: 78.98% for 'AEI'

to match random generated text, similar with the proposed expansion, and with letter occurrence probabilities as in the data analysis corpus.

By contrast, the accidental match probability (with the letter occurrence probabilities same as above) of 'ZEV' for "zero emissions vehicle" is quite low (0.96%), indicating a high likelihood of a correct match, even if the level of confidence is low that the acronym-expansion pair is a definition.

Accidental matches can be counterbalanced (as presented in chapter 5) in a two-phase machine learning approach through the use of evidence from large corpora such as the world wide web of the cooccurrence of the acronym and expansion.

6.2.4 Inversion

Acronym inversion (rule 2.14) is infrequent in English, but commonly occurs in other languages. The existence of inversion increases the ambiguity degree and accidental matching probability, since the restriction of direct order in the matches of acronym letters in the expansion is lifted.

Another consequence of acronym inversion is a potentially dramatic loss of efficiency. In a similar type (string alignment) of problem in molecular biology, present for instance in the secondary structure of nucleic acids, the computational complexity of solutions increases once a direct order restriction (the absence of so-called *knots*) is removed, from $O(n^2)$ to the complexity of the *Catalan number* (exponential). Even partial solutions to relatively simple special cases of knots in this domain have complexities of $O(n^3)$ to $O(n^5)$ [90].

A brute force approach which finds matches in conditions of inversion enumerates all permutations of the letters of an acronym and calculate a matching for each of them. This leads to an increase in complexity proportional to the factorial (exponential) of the size of the acronym. For long acronyms, such as syllabisms, this increase can be quite dramatic. In languages such as Russian, where inversion occurs systematically (hypothesis 2.5) and long acronyms are common, the computational complexity of the brute force approach makes it prohibitively expensive for large-scale implementations.

I submit here that acronyms generally only exhibit a restricted form of inversion:

Hypothesis 6.2 (Single-knot inversion) *Acronym inversion occurs at most with single knot crossing between groups of matches for all letters of the acronym with letters in the expansion, presented in the original order of words in the expansion.*

For example, the match between the expressions 'ABC' and 'ACB' has one single knot. If shown one on top of the other, and after connecting matching characters, only one crossing of connecting

lines occurs. All examples of inversion discussed so far exhibit single-knot inversion: (2.12), (2.32), (2.34), (2.33), (2.37), (2.81), (2.61) and (2.62).

A very simple change in the dynamic programming acronym-expansion matching algorithm which only matches in sequence (rule 2.13) can accommodate a strictly wider class of inversions than single-knot inversion: *match-twice*. For example, the acronym 'EFA' does not match consecutively the single expansion in (6.6) but *consecutively matches the duplicated expansion* in (6.7).

'EFA' no match "Europäisches Abkommen zum Schutz von Fernsehsendungen" (6.6)

'EFA' match "Europäisches Abkommen zum Schutz von Fernsehsendungen
Europäisches Abkommen zum Schutz von Fernsehsendungen" (6.7)

The loss in complexity for the matching the expansion twice is a constant factor, which is asymptotically insignificant.

6.2.5 Availability of Language-specific Linguistic Resources

The universal acronym formation rules are based on linguistically motivated hypotheses, and their use in specific applications is subject to the availability of linguistic resources for the target languages. The following such resources are needed:

- Inflectional morphology tools, which can help isolate prefixes, given the high productivity of morpheme matching (rule 2.2).
- Following similar reasoning, compound morphology resources, helping with the identification and splitting of morphological compounds, especially frequent in certain languages and domains.
- Syllabification tools, which can approximate morphological information, if morphological resources are missing.
- Lists of link words in the target language.
- Activations, ordering and weights of the rules in the universal set for the target language. These can only be determined experimentally.

6.3 Corpora

For this work, a multilingual list of acronyms from the European Union's Publication Office [36] is used, which contains lists of acronyms and abbreviations in eleven languages, loosely related to the European Community and current international affairs. Greek is excluded from the evaluation, and lists of acronyms definitions for the remaining ten languages are included in the *evaluation corpus*.

From each list of acronyms in a specific language in the evaluation corpus, all definitions of acronyms in other languages on the list are manually eliminated. Entries which are considered to be code names that are not acronyms are also eliminated. The following languages are represented: Spanish (es), Danish (da), German (de), English (en-a), French (fr), Italian (it), Dutch (nl), Portuguese (pt), Finnish (fi), and Swedish (sv).

From a dictionary of Russian acronyms and abbreviations [121], 300 entries are randomly chosen. Of these, all entries that are code names but not acronyms, and all entries that are direct abbreviations are manually eliminated. The resulting 245 entries are added to the *evaluation corpus* (ru).

The WWWAAS [145] corpus is also used, containing a total of 17,529 entries, most of them acronym-expansion pairs in the English language. This corpus represents a medium-size, manually built and maintained, general-purpose acronym dictionary, which is expected to provide a reasonable small-scale approximation for the body of all acronyms in English. From this corpus, 1,000 entries are chosen, which are the same (randomly selected) used earlier in the *testing corpus* from chapter 4.

From these entries, only the English language acronym-expansion pairs are chosen, and for each, all text surrounding the expansion is eliminated. The remaining 894 definitions form the *data analysis corpus* (en-b). Distributions of quantitative measures associated with acronym-expansion matches are evaluated on this corpus.

Another 335 acronym-expansion pairs in the English language are chosen at random from the WWWAAS corpus, and used as a *control corpus* (en-c), in order to provide a comparison between the *unilingual, generic* acronym dictionaries (such as WWWAAS) and the English language component of *multilingual, specific* acronym lists [36].

The resulting thirteen lists (da, de, en-a, en-b, en-c, es, fi, fr, it, nl, pt, ru, sv) in eleven languages are prepared for acronym-expansion matching.

For languages with rich compound morphology (Danish, German, Finnish, Dutch, Russian and Swedish), all words are manually split into morphemes and syllables and all link words are manually

marked.

For the English language data analysis corpus (en-b), all words are automatically split into morphemes, using the list of English inflexional prefixes used for the system presented in chapter 4. and further split into syllables, using Hammond's syllabifier, [50]. All link words are automatically marked using a list of prepositions, conjunctions, articles and particles in English.

For the remaining lists (English language evaluation corpus, English language control corpus, Spanish, French, Italian and Portuguese), all words are manually split into morphemes, and Hammond's syllabifier is used to automatically further split words into syllables. All link words are manually marked for each list.

For each list in the evaluation corpus, the control corpus and in the data analysis corpus, letter occurrence frequencies are collected, both for the whole document, as well as for initials of words only. Letter frequency information is further used to model accidental matching probabilities for each acronym-expansion pair in every list.

6.4 Regularities of Acronym-Expansion Pairs in the English Language

The acronym-expansion ambiguity degree and the accidental match probability is measured for all acronym-expansion pairs in the data analysis corpus.

The probability range (between 0:impossible and 1:certain) is split into intervals of size 0.05 and the number of acronym definitions whose accidental probabilities fall within each interval are counted. A linear regression approximation (method of least squares) is executed on the distribution of the logarithm of the range count. The lower bound of each interval is taken as the independent variable.

Figure 6.1 shows a chart of the distribution of the logarithm of number of occurrences of accidental match probability by interval, compared to its linear regression approximation. Observations with probability at or above 0.6 (the first time where observations touch the x-axis) are disregarded and indicated with empty circles. Good correlation ($R^2 = 0.9858$) between the logarithm of the count and the probability range is observed (significance $p < 0.001$). This indicates an exponential (Zipf-Mandelbrot) distribution of the density of accidental match probabilities.

The distribution of the ambiguity degree is measured by taking the logarithm of the ambiguity degree of each observation (log_{10}). All acronym-expansion pairs from the data analysis corpus with ambiguity degree falling within ranges of size 0.225, starting from 0 are counted. Figure 6.2 shows a chart of the distribution of ambiguity degree in logarithmic intervals, compared to its linear

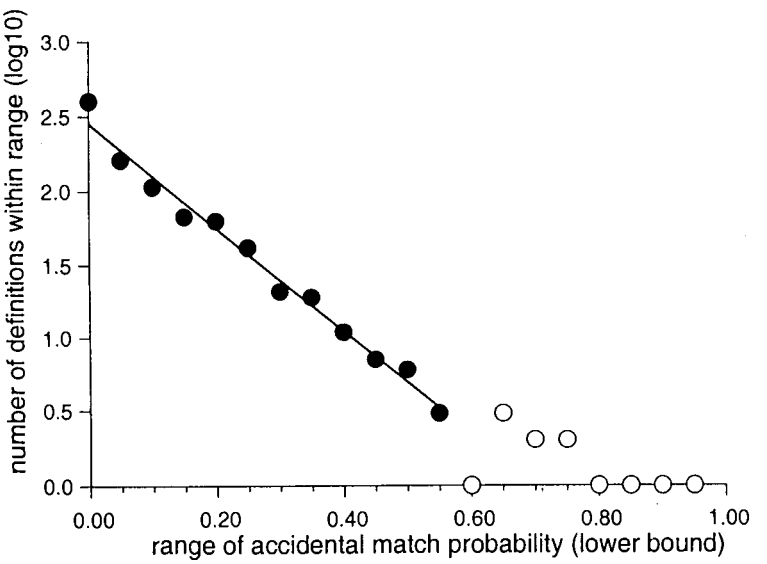


Figure 6.1: Distribution of accidental match probability in a corpus of acronym definitions (Figure by author)

regression approximation. Observations within ranges above 1.8 are disregarded and indicated with empty circles.

High correlation (R^2=0.9280) between the lower bound of each range and the definition count is observed (significance $p < 0.001$). This indicates similarity with an exponential (Zipf-Mandelbrot) distribution of the density of ambiguities of acronym-expansion matches.

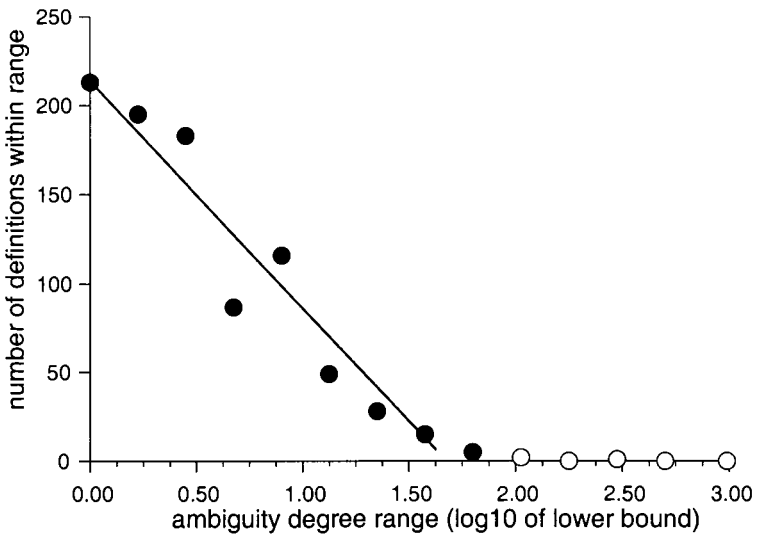


Figure 6.2: Distribution of acronym-expansion match ambiguity in a corpus of acronym definitions (Figure by author)

The results of dynamic programming for acronym-expansion matching presented in chapter 4, indicate support for the optimal substructure of the linguistic plausibility score of acronym-expansion matching in English, hypothesis 6.1.

6.5 Cross-Linguistic Regularities of Acronym-Expansion Pairs

All acronym-expansion pairs in the evaluation corpus are matched using ACE, a dynamic programming acronym-expansion matching algorithm presented in 4, using different weights for individual language groups. For individual languages, the order and weight of specific rules is indicated arrays

of ("rule number","status","rule weight") triples, with 'D' indicating a disabled rule, 'V' a violable rule and 'I' an inviolable rule.

The weights of matches for letters within groups of matches are calculated as *continuity* scores, the average between the weight of matching first letter in the contiguous section and the weight of matching the current letter.

Rules 2.9 (plural duplication), 2.10 (symbolic matching), 2.11 (migration) 2.12 (inflection) and 2.15 (import) are disabled. For rules with equal weights, preference is given to the first in the list.

For English, French and Italian, the ordering and weights of rules used by ACE are identical with the ones used in chapter4 (and the default for ACE):

{(2.1, V, capital:600; non-capital:450; capitalized link word: 400, link word:150), (2.2, V, 100), (2.3, V, 20), (2.4, V, average continuity score), (2.5, V, 10), (2.6, V, 100), (2.7, V, 100), (2.8, V, 1), (2.9, D, 0), (2.10, D, 0), (2.11, D, 0), (2.12, D, 0), (2.13, I, 0), (2.14, D, 0), (2.15, D, 0)}.

For other languages, the weights are adjusted after one run of ACE, to eliminate incorrect matches.

For Spanish, Portuguese and Danish:

{(2.1, V, capital:600; non-capital:400; capitalized link word: 500, link word:200), (2.2, V, 420), (2.3, V, 20), (2.4, V, average continuity score), (2.5, V, 10), (2.6, V, 100), (2.7, V, 100), (2.8, V, 1), (2.9, D, 0), (2.10, D, 0), (2.11, D, 0), (2.12, D, 0), (2.13, I, 0), (2.14, D, 0), (2.15, D, 0)}.

For German, Swedish, Dutch, Finnish and Russian:

{(2.1, V, capital:600; non-capital:450; link word:150), (2.2, V, 420), (2.3, V, 20), (2.4, V, average continuity score), (2.5, V, 10), (2.6, V, 100), (2.7, V, 100), (2.8, V, 1), (2.9, D, 0), (2.10, D, 0), (2.11, D, 0), (2.12, D, 0), (2.13, V, 0), (2.14, V, 0), (2.15, D, 0)}.

The lists are grouped into four categories: English, Latin languages (French, Italian, Portuguese, Spanish), Germanic Languages (Dutch, Danish, Swedish, German) and Other (Finnish, Russian).

The following values are measured for each list:

- Number of acronym-expansion pairs.
- Number of characters per acronym: average, standard deviation and maximum.
- Number of characters per expansion: average, standard deviation and maximum.
- Number of words in the expansion matched by letters in the acronym: average, standard deviation and maximum per acronym-expansion pair.

- Accidental match probability, calculated using equation (6.5): average and maximum per acronym-expansion pair. For each list, letter occurrence counts in expansions in the list are used to approximate letter occurrence probabilities.
- Ambiguity degree, calculated using equation (6.4): average and maximum per acronym-expansion pair.
- Average number letters per acronym which match in the expansion: word initials, intra-word morpheme initials and, respectively, internal letters which belong to group matches.
- Average number of morphemes per expansion and per word.
- Total numbers (per list) of occurrences of inversion, inflection and migration.
- Expected occurrence within an acronym-expansion pair (calculated as total number of occurrences over number of acronym-expansion pairs) of inversion, inflection and migration.
- Total number (per list) of acronym-expansion pairs for which ACE reports correct, incorrect and "not found" results.
- Precision, recall and $F_{\beta=1} = \frac{2 \times \text{precision} \times \text{recall}}{(\text{precision} + \text{recall})}$ of acronym-expansion matching using ACE, calculated per list.

Result are presented in Table 6.5, and observations are discussed below.

All lists contain reasonably many examples (over 200 acronym-expansion pairs), with the exception of Finnish (whose list is smallest, with only 58 entries) and Germanic languages other than German. The English language control corpus en-c is chosen in such a way that its size (235 entries) is close to the size of the English language evaluation corpus (231 entries).

The values for average acronym length (in characters): (3.25 to 3.96) $\pm$ (0.6 to 1.22) are consistent across all lists, with the exception of Russian, where the presence of syllabisms accounts for longer acronyms and higher variability of the length: 4.37 ± 3.88 characters.

The average lengths in characters of expansions are also consistent across lists, slightly longer for Latin languages and the English evaluation corpus.

The average number of words in expansions matched by letters in acronyms is very close to the average number of matches at initials, indicating that the vast majority of words in expansions that are matched by letters in the acronym *match at least at the initial*, across all lists. The average

number of matched initials is consistent across languages in the Latin group and English, and slightly higher than for languages in the Germanic group, Russian and Finnish.

The average number of matches in groups is also consistent across lists, with the exception of Russian, where large group matches within syllabisms account for an increase in the average number of group matching letters to almost one per expansion. French, Danish, Spanish and the English evaluation corpus have also higher group matching averages than the rest. The difference for the two English language corpora (0.13 in en-c vs. 0.28 in en-a) can be explained by the bias of the evaluation corpus (European Community and current international affairs).

The average number of morpheme matches is significantly higher for Germanic languages and Finnish, due to their productive morphology. A clear preference to match on intra-word morpheme initials occurs for Germanic languages and Finnish: around half of the morphemes are matched, compared to around 10% for the other languages, and about one third for Russian.

The accidental match probabilities are reasonably close between all lists, with the exception of the English language control corpus. The ambiguity degree is higher for Latin languages and the English evaluation corpus.

The highest average incidence per acronym-expansion pair of acronym inversion occurs in German (1.99%), Russian (3.27%) and Finnish (3.45%, although just for two related instances). All cases of inversion are single-knot inversion (hypothesis 6.2), and are solved through the "match-twice" application of the ACE dynamic programming matching algorithm.

Most of the cases of inversion are morphological (hypothesis 2.5), with the exception of one case in German and all three in English of accidental inversion (hypothesis 2.5).

Migrations only occur in Russian, and one case in Portuguese ('ç' to 'C'). Inflections occur only in Russian. Since all migration and inflection rules are disabled, the system fails for these; incorrect matches are generated for all. Other matches occur incorrectly within groups, rather than at initials of noise words (in English) but their incidence is too low to draw conclusions.

All three acronym-expansion pairs for which matches are not found are cases of symbolic matching, such as the debatable occurrence in English of 'P9' for 'pine'.

6.6 Summary and Conclusions

Difficulties related to the automated acquisition of acronyms from text are quantified through the definition of the measures: ambiguity and accidental matching probability. A hypothesis (6.1) of

	English		Latin Languages				Germanic Languages				Other	
	en-c	en-a	fr	it	pt	es	nl	da	sv	de	fi	ru
acronym-expansion pairs	335	331	496	232	516	431	131	97	85	352	58	245
characters per acronym (avg.)	3.61	3.96	3.88	3.59	3.67	3.78	3.31	3.28	3.25	3.36	3.33	4.37
characters per acronym (st.dev.)	1.09	1.22	1.13	0.93	1.05	1.13	0.83	0.99	0.6	0.82	1.11	3.88
characters per acronym (max.)	8	10	8	7	8	8	7	8	5	7	7	22
characters per expansion (avg.)	26.21	32.35	36.35	33.72	33.67	33.97	29.66	27.87	29.05	29.87	25.6	26.77
characters per expansion (st.dev.)	10.76	12.42	13.88	12.93	11.74	11.94	9.03	8.68	9.31	11.87	9.45	17.58
characters per expansion (max.)	79	81	110	113	94	75	58	61	63	88	46	126
words matched (avg.)	3.4	3.6	3.59	3.46	3.51	3.49	2.75	2.25	2.39	2.22	2.41	2.59
words matched (st.dev.)	1.01	1.01	0.99	0.83	0.93	0.92	0.84	0.87	0.76	0.99	0.92	1.2
words matched (max.)	7	9	8	6	6	7	5	4	4	5	4	8
accidental match prob. (avg.%)	9.87	16.23	20.58	19.93	24.43	22.48	16.35	16.70	18.06	11.90	17.12	14.42
accidental match prob. (max.%)	75.60	95.24	95.40	82.94	92.81	86.44	62.94	76.95	66.33	79.49	63.20	68.70
ambiguity degree (avg.)	7.94	24.19	30.05	24.91	19.97	20.92	7.1	6.65	7.05	4.92	4.1	13.08
ambiguity degree (max.)	120	1419	1305	2123	1788	888	50	71	91	56	16	888
initial matches (avg.)	3.39	3.6	3.58	3.45	3.5	3.49	2.75	2.25	2.39	2.21	2.41	2.54
morpheme matches (avg.)	0.03	0.03	0.04	0.02	0.04	0.05	0.42	0.81	0.78	0.89	0.5	0.13
matches in groups (avg.)	0.13	0.28	0.24	0.09	0.1	0.21	0.11	0.22	0.07	0.14	0.14	0.91
morphemes per expansion	0.23	0.35	0.27	0.26	0.33	0.37	1.11	1.54	1.33	1.86	1.24	0.44
morphemes per word	0.06	0.07	0.05	0.05	0.07	0.07	0.32	0.51	0.48	0.64	0.46	0.13
inversions	1	2								7	2	8
inflections												4
migrations					1							2
expected inversions (%)	0.3	0.6								1.99	3.45	3.27
expected inflections (%)												1.63
expected migrations (%)					0.19							0.82
correct	335	329	495	232	514	431	131	97	85	352	58	239
incorrect		2	1		2							6
not found	1					1						1
precision (%)	100	99.4	99.8	100	99.61	100	100	100	100	100	100	97.55
recall (%)	99.7	100	100	100	100	99.77	100	100	100	100	100	99.58
$F_{\beta=1}$(%)	99.85	99.7	99.9	100	99.81	99.88	100	100	100	100	100	98.56

Table 6.1: Results of acronym-expansion matching on the evaluation corpus (Table by author)

optimal substucture of acronym-expansion matching is submitted, which avoids computational complexity issues related to ambiguity. Accidental matching probability is used chapter 5 as a training feature for acronym acquisition.

The ambiguity and accidental match probability is evaluated on a corpus of 824 acronym-expansion pairs in the English language. The densities of these measures are found to follow closely Zipf-Mandelbrot distributions, indicating that acronym-expansion matching difficulty is low in most cases, with a trail of increasingly difficult examples extending, similarly with other sparse data problems.

Acronym inversion is another difficulty of acronym acquisition, through an increase in the probability for spurious matches and a decrease in efficiency. A hypothesis (6.2) is submitted that restricts the types of inversion allowable, resulting in no asymptotic loss of performance.

The availability of language-specific linguistic resources (morphological, lexical) are prerequisites to acronym-expansion matching components of acronym acquisition systems.

A dynamic programming algorithm, ACE, is used to match acronym-expansion pair in eleven languages, from three distinct corpora. The results of the evaluation are excellent, with $F_{\beta=1}$=98.58% for Russian (due to the lack of coverage for acronym inflection and migration) and $F_{\beta=1}$ >99% for all other languages. The results indicate the adequacy of the universal theory of acronyms for the studied languages.

Comparison between languages shows quantitative differences in morpheme matching (based on predominance of morphological compounds), ambiguity degree, group matching, inversion, inflection and migration.

Hybrid automatic acronym acquisition methods based on the algorithm ACE have been proven in chapter 5 to achieve good performance on noisy English language text, after training on matching adequacy features (generated by the algorithm) and acronym-expansion pair cooccurrence information extracted from large text collections accessed through web search engines. This work indicates that these methods of automatic acronym acquisition should also be directly applicable to other languages, at least to the ones covered in this study.

This work shows that automatic acronym acquisition is of a level of difficulty similar with English in languages other than Russian, and more difficult in Russian, due to the high incidence of migration and inflection phenomena.

Chapter 7

Automatic Acronym Sense Disambiguation

The polysemy of acronyms bears similarities with that of common words. This chapter presents an automatic methodology for the disambiguation of acronym senses, which uses unsupervised machine learning, and starts from an acronym sense dictionary.

Training data is automatically extracted from downloaded documents identified from the results of search engine queries. The resulting system is automatically evaluated on a number of acronym forms with multiple corresponding expansions in the English language and achieves accuracy over 92.58% at picking the correct sense for an acronym in a given document and $F_{\beta=1}$=91.52% at deciding whether a given sense of an acronym matches a given occurrence.

7.1 Acronyms and word sense disambiguation

Acronym sense disambiguation has been studied in the medical domain by [112], using a supervised machine learning system, and achieving a good level of success (89% accuracy) but has so far been an open problem for general text.

Acronym sense disambiguation represents a special case of the more general problem of Word Sense Disambiguation (WSD), one of the most difficult and elusive open problems in Natural Language Processing.

The last decade has seen a lot of interest directed to WSD, with a special issue of the ACL

journal, *Computational Linguistics* [63], two combined special issues of *Computers and the Humanities*, [71], two special issues of the *Journal of Natural Language Engineering*, and also the first two instances of the SENSEVAL[1] multilingual word sense disambiguation system evaluation events.

In spite of this high level of interest, I argue that the state of the art has reached a level of stagnation, as can be illustrated by the modest advancement (3.84%) in precision and recall reported to baseline (as described in section A.6.4) between the best performing systems at SENSEVAL-1 and SENSEVAL-2 on the "English Lexical Sample - Coarse-grained Scoring" task.

The main difficulties of WSD lie in the fluid definition of *word sense*: as an *intrinsic property* of certain words, as *possible uses of a given word in context* [39], of different *lexical granularity* [148] and with possible *simultaneous activations* [53]. In practice, boundaries with concepts such as *metaphor*, *metonymy* and *synecdoche* are blurry at best.

Although it is generally agreed that WSD must rely on some form of machine learning, the high costs of acquiring consistent sense repositories and creating training sets of adequate coverage are significant obstacles ahead, as is the high cost of conducting unbiased large-scale evaluations.

It is proposed here that general WSD difficulties either do not exist at all for acronym sense disambiguation, or can be surmounted through methods shown further, using data and resources readily available on the Internet.

7.2 The polysemy of acronyms

Acronym senses correspond precisely (one-to-one) to expansions. The terms *sense* and *expansion* will be used interchangeably throughout this document, when referring to acronyms. The relationship between expansions and acronym forms is often many-to-one, with one acronym form having multiple senses, in different documents, or contexts. Compared to senses of common words, an acronym sense is very precise, and the acronym is predominantly used only as an exact placeholder for its expansion.

The polysemy of acronyms is a very significant issue, as can be illustrated by the 64 different expansions (senses) for the acronym form 'CIA' shown by Acronym Finder in October 2003.

Hypothesis 7.1 *The polysemous distribution of acronyms is similar with that of regular words. The polysemy of acronyms is at least as productive as that of regular words.*

[1] On the web http://www.senseval.org (February 2004).

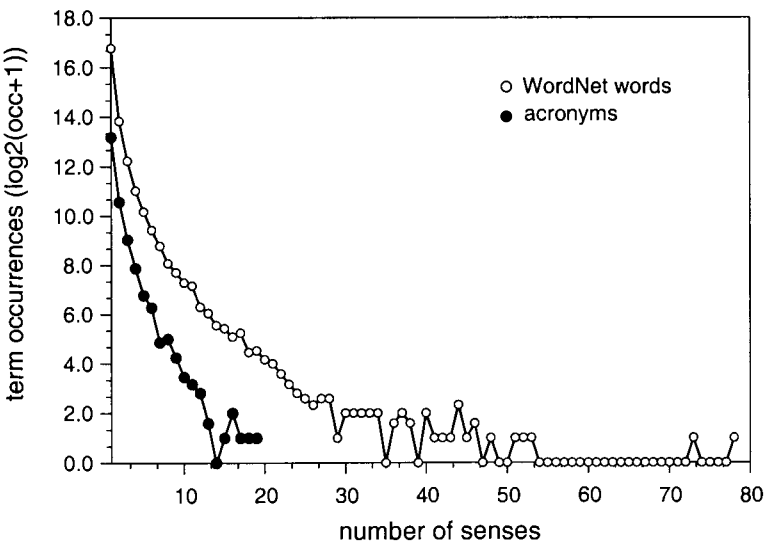


Figure 7.1: Distribution of term senses for WordNet words and acronyms from the WWWAAS database (Figure by author based on data from WordNet and WWWAAS)

The distributions for number of senses of words from WordNet (including part-of-speech variations, a total of 136,972 senses) and of acronyms from the WWWAAS database (a total of 16,823 acronyms) are plotted in figure 7.2.

The results illustrate similarities and differences between acronym senses and word senses; most terms have one sense: 81.72% for WordNet, 52.03% for acronyms. Acronyms are more polysemous than words. I submit here that this is partly due to the fact that acronyms are shorter on average ($\approx$ 3.71 characters) than regular words ($\approx$ 7.84 characters), resulting in an increased likelihood of collisions. The number of terms with a given number of senses decreases with the number of senses.

For our sample databases, the maximum polysemy of words (the word 'break' has 78 senses in WordNet) is higher than the maximum polysemy of acronyms (the acronym 'AACA' has 18 senses), but that can be explained through the smaller sample size; moreover, if senses were randomly taken out of WordNet up to reaching the size of the WWWAAS database, the word 'break' would be expected to occur only $\frac{16,823}{136,972} \times 78 \approx 9.58$ times.

Unlike common words, acronym senses are generally unrelated and non-overlapping. *False polysemy*, such as the occurrences of the two equivalent senses of 'MIME', as "*Multimedia* Internet *Message* Extensions" (RFC-1590, http://ietf.org, February 2004), or "*Multipurpose* Internet *Mail* Extensions" (RFC-1524, http://ietf.org, February 2004) are uncommon and their resulting semantic difference is – arguably – irrelevant.

Such occurrences represent *sense interaction* within documents or domains, which increase the difficulty of disambiguation. Sense interaction is also present in situations outside false polysemy, such as in the example of 'ABC', an acronym form for both "American Broadcasting Company" and "Australian Broadcasting Corporation", both of which represent television networks with national coverage in the USA and Australia, respectively. Given the very similar contexts (news stories, corporate announcements within the news industry, a common pool of professionals and news sources, etc) of the use of these acronyms in text, I submit that even humans need to use non-textual (pragmatic) information when differentiating between them, if at all able to do it.

A measure of sense interaction for pairs of senses of an acronym, in general, even in the absence of any training information, is the *normalized perplexity of cooccurrence* within web documents of the phrases e_1 and e_2, representing the expansions (senses). This value can be calculated using individual search engine occurrence hit counts of the two expansions (O_1 and O_2), the cooccurrence hit count (O_{12}), a system constant indicating the number of acronym lists available online ($L = 17$)

and the total number of documents indexed by the search engine (e.g. Google) $G \approx 3 \cdot 10^{10}$:

$$\pi(e_1, e_2) = \log_{10} \frac{\max\{O_{12} - L, 0\} \cdot G + 1}{O_1 O_2 + 1}$$

For example, the phrases ABC_1 ="American Broadcasting Company" and ABC_2 ="American Broadcasting Corporation" (arguably a case of false polysemy) have a perplexity of cooccurrence $\pi(ABC_1, ABC_2)$=3.84, in other words, the probability of their cooccurrence within a document is $10^{3.84} \approx$ 6,918 times greater than chance.

Similarly, the phrases ABC_1 ="American Broadcasting Company" and ABC_3 ="Australian Broadcasting Corporation" (a case of true sense interaction) have a perplexity of cooccurrence $\pi(ABC_1, ABC_3)$=3.10, in other words, the probability of their cooccurrence within a document is $10^{3.03} \approx$ 1,071 times greater than chance.

For situations where significant interaction can be determined between senses e_1 and e_2 of an acronym, a decision not to disambiguate between these senses can be made based on $\pi(e_1, e_2) \geq \overline{\pi}$, where $\overline{\pi}$=3.0 is a system constant. In other words, if a system returns one of the senses e_1, e_2, return both to the user.

7.3 Acronym disambiguation system

A system that uses machine learning based on pattern recognition is proposed, using support vector machines (SVM): a statistical learning technique based on the work of [139]. SVM have been successfully used for many natural language processing tasks, such as chunking [80], text classification [65], and acronym acquisition (chapter 5).

Support vector machines are used for classification tasks, in most situations in a *pattern recognition* configuration, in which the result of testing is a yes/no answer to the question whether a previously unseen example belongs to (fits the pattern of) a given group of examples. *Values vectors* are multi-dimensional vectors in feature space, and are represented as sequences of feature-value pairs. Each feature is a positive integer, and each value is a real number, in floating point representation.

SVM training is performed on sets of value vectors, where each vector is either a positive example (labeled $+1$), or a negative example (labeled -1). The *positive emphasis* ϵ^+ is a training parameter, which represents the factor by which positive examples are desired to outweigh negative examples. A positive emphasis $\epsilon^+ > 1$ is useful when the training set consists of predominantly

negative examples.

The result of SVM training is a *classification model*, which is used during the SVM testing phase to *classify* other value vectors. The result of classification is a *decision value* ($\in \mathbb{R}$) for each vector. A positive (including 0) decision value means that the vector was classified as fitting the pattern and a negative value means that the vector was classified as not fitting the pattern.

Features are terms occurring in the same document as the target acronym form. The terms are either words that occur in WordNet [37] or other acronym forms, following assumption 7.2.

Hypothesis 7.2 *The presence of specific acronym forms in a document can be a good sense indicator for other polysemous words.*

The impact of common words is minimized by requiring a minimum length for WordNet word features. This is intuitively adequate, following Zipf's law.

One single value ($\delta_D(t, A)$) is used to model both the presence of a given term (t) in the same document (D) as the target acronym (A) and the distance in words (d) between occurrences of the term (t_i) and occurrences of the target acronym (A_j).

$$\delta_D(t, A) = \begin{cases} \alpha + \dfrac{\beta}{\min\limits_{t_i, A_j \in D} d(t_i, A_j)} & \text{if} \quad t \in D \\ 0 & \text{if} \quad t \notin D \end{cases}$$

The values for the system-wide constants $\alpha = 0.25$ and $\beta = 1.0$ are chosen in a way in which the presence of a given term in the same document as the target acronym accounts quantitatively as much as the proximity within a four-word window around each target acronym occurrence. The value of $\delta_D(t, A)$ degrades smoothly with the increase of the distance between the target acronym and specific terms.

For each document-acronym form pair, the values for $\delta_D(t, A)$ are calculated and collected into one feature vector. For each acronym sense, the system is trained on collections of documents (value vectors) with known senses. Positive examples are documents (and associated value vectors) containing occurrences of the acronym form with the target sense, and negative examples are value vectors corresponding to documents where the acronym form occurs with other senses.

The positive emphasis is calculated for each acronym sense (S_A) from the number of documents, $N(S_A)$, containing the acronym form A with the sense S_A and the number of documents, $N(\overline{S_A})$,

containing other senses of the acronym form A :

$$\epsilon^+(S_A) = \left\lceil \frac{w^+ \times N(\overline{S_A})}{N(S_A)} \right\rceil$$

The value $w^+ = 2$ is a system constant, chosen to favor false positive errors compared to false negative errors (increase recall at the expense of precision).

The result of training is a set of *classification models*, one for each sense of a given acronym. Feature vectors are calculated for documents with occurrences of acronym forms with unknown senses. Disambiguation is performed by two tasks:

- *Decision*: a boolean answer to the question whether a given acronym occurrence is associated with a given sense.
- *Selection*: the selection of one of a list of senses for a given acronym form occurrence.

The decision task is based on the result of "pattern recognition" using the learned classification model for the acronym sense, applied to the target document's feature vector.

The selection task identifies the maximum value of the decision function resulting from the decision task applied repeatedly to the value vector of the target document, using consecutively the classification models for each sense.

To illustrate, consider an acronym form X, with n senses: $s_1..s_n$. The training set for senses of the acronym form X is a set of value vectors, V_X. For each sense s_k ($k \in 1..n$), separately, all vectors $u \in V_X$, with Sense(u) $= s_k$ are considered positive examples (labeled $+1$). All other vectors in V_X are considered negative examples (labeled -1). A classification model for s_k is obtained through SVM training on V_X such labeled, and is used to classify a new vector v, resulting in a classification value $c(s_k, v)$. For each sense, the result of the classification decision is given by:

- Sense(v) $= s_k$ if $c(s_k, v) \geq 0$
- Sense(v) $\neq s_k$ if $c(s_k, v) < 0$

The result of selection is given by:

$$\text{Sense}(v) = s_k \text{ where } k = \arg_k \max_{1 \leq k \leq n} c(s_k, v)$$

7.4 Acquisition of Training Data

Hypothesis 7.3 *Acronym forms occur generally with one sense per document, similar to the* one word sense per discourse *heuristic [147].*

Exceptions are represented by documents which are, or contain, wide coverage lists of acronyms. Those are removed by requiring that each document taken into account contains at most a given number of acronym forms, a system constant ($\gamma_A = 100$). A limitation of the size in tokens of each document (a system constant $\gamma_t = 10{,}000$) is also imposed, based on the assumption that very large documents (e.g. whole books) will induce noisy input during training.

Hypothesis 7.4 *The cooccurrence in a given document of an acronym form and the complete phrase of a corresponding acronym expansion indicates the sense of the acronym form to be the acronym expansion.*

For example, the cooccurrence in a document of the acronym form 'SPS' and one of its expansions "solar power satellite" indicates the sense "solar power satellite" for all occurrences of 'SPS' within the document.

Locating documents containing both an acronym and its expansion is relatively simple, as most search engines support queries containing contiguous phrases. For example, a query to search for documents containing a given multi-word phrase can be issued to Google by including the phrase within quotation marks.

HTML web documents are highly interconnected, and links to additional information regularly overshadow textual content. Hits of organizational or concept portals dominate many times the first pages of results returned by search engines. For example, a search for 'CIA "central intelligence agency"' will return the home page for 'CIA', which is a portal (a collection of links) to sources of information about various aspects of the CIA. On that page, the acronym 'CIA' will occur predominantly as a heading to normally flowing text, rather than as a term within the text.

Hypothesis 7.5 *Web documents in the Adobe PDF format are more likely sources of natural occurrences of specific acronyms in text than HTML documents.*

A number of search engines, including Google, have started to index documents in file formats other than HTML (e.g. PostScript, Adobe PDF, pure text, etc.). Following assumptions 7.4 and 7.5, a training corpus is built from English language documents in the Adobe PDF format, which contain occurrences of both the expansion (as a contiguous phrase) and the acronym.

Links to such documents available on the web can be obtained through search engine queries. The following is an example of one such Google query:

SPS "solar power satellite" filetype:pdf

For each acronym-expansion pair, 100 documents (or at least as many as available in the result of the query) are downloaded. Each document is converted to text.

In each of the resulting (successfully converted) text documents, all occurrences of the expansion are replaced with the acronym form (in the example above all occurrences of "solar power satellite" are replaced with 'SPS') obtaining documents where the target acronym occurs with a given sense, and without an explicit indication of its expansion. Each document is marked with the specific acronym sense.

The result of the training data acquisition phase is a training corpus, containing a training set for each acronym, with documents containing acronym forms, but not explicit acronym expansions, and labeled using the senses (expansions) of the acronym.

7.5 Evaluation

Evaluation results of word sense disambiguation systems (or any other NLP system) are inevitably biased by the choice of training and evaluation corpora. Since the system presented here does not rely on human markers, leave-one-out cross validation is performed on all data from the training corpus.

For each acronym-sense pair, every document (feature vector) containing the acronym form in the training corpus is successively isolated into a testing vector. Classification vectors are learned from all other documents (feature vectors) and are used to classify the testing vector. In other words, the system attempts to infer the correct sense for the acronym form in each document of the training corpus, using only examples from the other documents in the training corpus.

For example, consider the acronym form 'ABC' which occurs within our acronym database with seven senses: $ABC_1..ABC_7$, one of them being ABC_4 ("Alberta Blue Cross").

A training set of up to 100 documents per sense is downloaded and converted from Adobe PDF format to text. Each document contains occurrences of 'ABC' and the complete contiguous phrase of the expansion (sense). All occurrences of all expansions of 'ABC' in the training set are replaced with 'ABC'. All documents within the training set are labeled with their appropriate sense (the expansion that occurred within the document, before the replacement).

For every document D from the training set, in which 'ABC' occurs with the sense ABC_4, a model for the classification of occurrences of 'ABC' into ABC_4 and not ABC_4: $\overline{ABC_4}$ (decision) is created through training on all documents within the training set, with the exception of D. The model is tested on D and results in success if D is classified as containing ABC_4, and failure otherwise.

For every document D from the training set, classification models are also created for all other senses (ABC_i, where $i \neq 4$) through training on all documents in the training set, with the exception of D. All documents containing ABC_i are labeled as positive examples and all other documents (including documents containing ABC_4) are labeled as negative examples. The classification scores for testing D against all such classification models are collected. If the classification score for ABC_4 is the maximum, this indicates success on the selection task; failure is reported otherwise.

Results are consolidated for every sense and for every document. The approach is unbiased, since data in the training corpus was not seen before the end of the evaluation.

For the decision task, *precision*: p (correct classification decisions over positive classification decisions) , *recall*: r (correct classification decisions over number of positive examples) and $F_{\beta=1} = \frac{2pr}{p+r}$ are measured for each acronym sense.

For the selection task, *accuracy* (correct sense selections over number of documents) is measured for each acronym form.

The WWWAAS database contains 825 acronym forms with at least three associated senses; 150 of these are randomly selected, and training sets are built for each of them, from 100 downloaded Adobe PDF documents per sense, or at least as many as available through Google queries. Documents that cannot be converted to text, and which exceed the γ_A and γ_t restrictions are eliminated. Senses associated with less than ten qualifying documents are eliminated, and acronyms which consequently end up with less than three senses are also eliminated. The resulting 9,963 documents, distributed between 167 senses of 47 acronyms represent the *evaluation corpus.*

Leave-one-out cross validation (as described above) is performed for each of the acronyms in the evaluation corpus. A data flow diagram describing the operation and evaluation of the system is shown in Figure 7.5.

The results are consolidated in global performance decision and selection figures: $F_{\beta=1}$ = 91.52% (precision: 90.57%, recall: 92.48%) and accuracy 92.58%. Baseline $F_{\beta=1}$ and accuracy are calculated for always choosing the most frequent sense and are both 36.94%. Evaluation results for each acronym form are reported in Table 7.1. Performance figures for ten other acronyms, used during the development of the system are not reported.

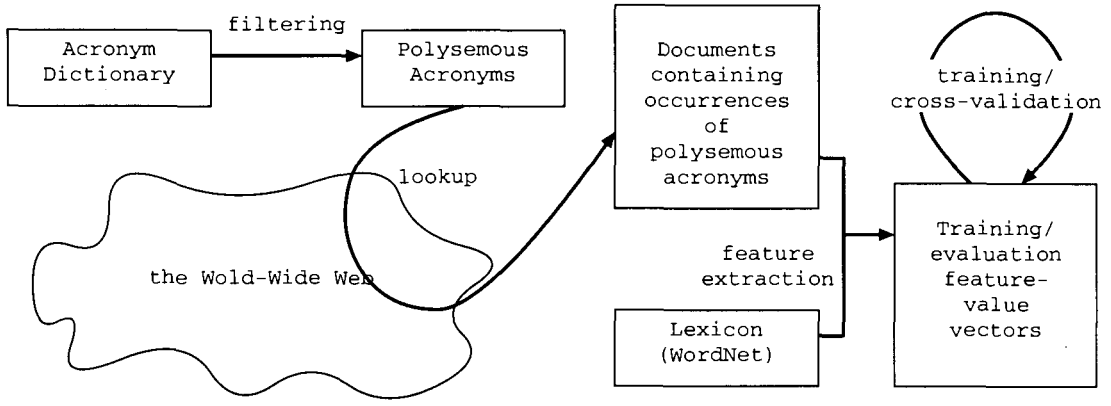


Figure 7.2: Data flow within the acronym disambiguation system, including evaluation (Figure by author)

Cooccurrence perplexity is calculated for all pairs of senses of each acronym form. For each acronym, all pairs e_1, e_2 of senses for which $\pi(e_1, e_2) \geq 3$ are grouped together. When not trying to differentiate between senses in such groups, the resulting accuracy increases to 97.14% and the $F_{\beta=1}$ increases to 95.08% (precision : 96.88%, recall: 93.35%). Acronyms for which all senses interact (baseline accuracy after sense interaction 100.00%) are not included in the figures reported after sense interaction. 'DEC', 'EOT', 'IFF' and 'SAP' are such examples.

Baseline $F_{\beta=1}$ and accuracy for picking the most frequent sense or group (when not trying to differentiate between members of the same group) from the training corpus are both 48.45%.

The following external components are used: *SVMlight*: an SVM implementation [66]; the GNU Ghostscript utility *ps2ascii*; the *Google Java API*, for executing automated bulk queries to the Google search engine; and the GNU *wget* utility, for bulk downloads of web documents.

Table 7.1: Results of acronym disambiguation on the evaluation corpus. 'Acr':acronym; 'co':correct; 'in':incorrect; 'nf':not found; 'pr':precision; 'rec':recall; 'F':$F_{\beta=1}$; 'acc':accuracy. Accuracy and baseline accuracy after consolidating interacting senses are in *italic* and $\overline{\textit{overline italic}}$ respectively. Interacting senses for each acronym are in *italic*. Distinct groups of interacting senses for the same acronym form are indicated with ◇ and • (Table by author)

Acronym	Expansion	co	in	nf	pr	rec	F	acc
ACE	advanced composition explorer	82	2	3	97.62	96.47	97.04	95.43
	agricultural communicators in education	55	0	7	100.00	88.71	94.02	
	american council on education	71	3	6	95.95	92.21	94.04	
	angiotensin converting enzyme	80	2	6	97.56	93.02	95.24	
	automatic calling equipment	6	1	5	85.71	54.55	66.67	
	automatic computing engine	23	0	6	100.00	79.31	88.46	
ANC	active noise control	70	3	2	95.89	97.22	96.55	97.14
	african national congress	92	8	0	92.00	100.00	95.83	
	army nurse corps	43	1	3	97.73	93.48	95.56	
ANS	advanced network and services	25	0	2	100.00	92.59	96.15	91.87
	american nuclear society	71	12	1	85.54	98.61	91.61	
	audubon naturalist society	20	0	4	100.00	83.33	90.91	
APA	all points addressable	46	0	3	100.00	93.88	96.84	96.04
	american philosophical association	76	4	3	95.00	96.20	95.60	
	asian pacific american	73	7	1	91.25	98.65	94.81	
ATIS	aids treatment information service	70	0	1	100.00	98.59	99.29	97.39
	automatic terminal information service	66	2	0	97.06	100.00	98.51	
	automatic transmitter identification system	11	0	5	100.00	68.75	81.48	
BOM	beginning of message	50	0	8	100.00	86.21	92.59	90.07
	bill of materials	67	14	1	82.72	98.53	89.93	
	book of mormon	20	0	5	100.00	80.00	88.89	
BRA	baseline risk assessment	55	0	2	100.00	96.49	98.21	98.50
	basic rate access	70	0	1	100.00	98.59	99.29	
	boston redevelopment authority	72	1	0	98.63	100.00	99.31	
CAI	community association institute	39	0	3	100.00	92.86	96.30	67.38
	computer aided instruction	63	55	10	53.39	86.30	65.97	*97.86*
	computer assisted instruction	61	59	11	50.83	84.72	63.54	$\overline{77.01}$

Table 7.1: Results of acronym disambiguation on the evaluation corpus (continued)

Acronym	Expansion	co	in	nf	pr	rec	F	acc
CAS	chemical abstracts service	80	4	0	95.24	100.00	97.56	90.91
	column address select	56	9	9	86.15	86.15	86.15	*98.70*
	column address strobe	72	17	2	80.90	97.30	88.34	*64.50*
	computer aided styling	7	3	5	70.00	58.33	63.64	
CBR	case based reasoning	87	0	3	100.00	96.67	98.31	98.34
	constant bit rate	83	1	1	98.81	98.81	98.81	
	cosmic background radiation	64	1	3	98.46	95.52	96.97	
CCR	commitment concurrency and recovery	47	0	1	100.00	97.92	98.95	95.78
	covenants conditions and restrictions	87	14	0	86.14	100.00	92.55	
	creedence clearwater revival	28	10	3	73.68	90.32	81.16	
CEC	commission of the european communities	88	4	1	95.65	98.88	97.24	96.70
	contractor establishment code	16	1	5	94.12	76.19	84.21	
	cooperative engagement capability	68	1	4	98.55	94.44	96.45	
CFP	call for papers	73	3	3	96.05	96.05	96.05	97.34
	certified financial planner	76	3	0	96.20	100.00	98.06	
	computers freedom and privacy	33	0	3	100.00	91.67	95.65	
CLI	*call level interface*	71	2	5	97.26	93.42	95.30	94.83
	clear interrupt	49	3	4	94.23	92.45	93.33	*100.00*
	command line interpreter	72	5	9	93.51	88.89	91.14	*90.52*
	current law index	20	1	2	95.24	90.91	93.02	
COB	chip on board	69	2	2	97.18	97.18	97.18	97.07
	close of business	87	5	2	94.57	97.75	96.13	
	coordination of benefits	74	1	5	98.67	93.67	96.10	
CPA	canadian paraplegic association	36	3	3	92.31	92.31	92.31	92.02
	certified public accountant	79	4	1	95.18	98.75	96.93	
	communications and public affairs	36	4	8	90.00	81.82	85.71	
CPC	certified personnel consultant	20	0	2	100.00	90.91	95.24	94.21
	circuit provisioning center	9	0	4	100.00	69.23	81.82	
	communist party of china	60	0	2	100.00	96.77	98.36	
	cost per copy	22	5	2	81.48	91.67	86.27	
CTC	*canadian transport commission*	34	2	9	94.44	79.07	86.08	93.50
	centralized traffic control	74	4	3	94.87	96.10	95.48	*99.50*
	cyclists touring club	79	11	1	87.78	98.75	92.94	*60.00*

Table 7.1: Results of acronym disambiguation on the evaluation corpus (continued)

Acronym	Expansion	co	in	nf	pr	rec	F	acc
DAC	*data acquisition and control*	70	9	6	88.61	92.11	90.32	96.02
	design automation conference	68	2	2	97.14	97.14	97.14	*98.67*
	digital to analog converter	76	9	4	89.41	95.00	92.12	*69.03*
DCE	data circuit terminating equipment	62	37	15	62.63	80.52	70.45	83.44
	data communications equipment	65	37	17	63.73	79.27	70.65	*100.00*
	dichloroethene	74	2	0	97.37	100.00	98.67	*76.88*
	distributed computing environment	84	2	3	97.67	96.55	97.11	
DEC	*decrement*	58	6	4	90.62	93.55	92.06	89.33
	device clear	7	0	11	100.00	38.89	56.00	*100.00*
	digital equipment corporation	66	8	4	89.19	94.29	91.67	*100.00*
DSC	differential scanning calorimeter	69	4	0	94.52	100.00	97.18	98.27
	digital selective calling	77	0	0	100.00	100.00	100.00	
	document structuring conventions	82	0	3	100.00	96.47	98.20	
DSS	decision support systems	71	0	9	100.00	88.75	94.04	95.70
	department of social services	74	1	4	98.67	94.87	96.73	
	digital satellite system	65	4	9	94.20	87.84	90.91	
	digital signature standard	77	2	2	97.47	97.47	97.47	
	digital spread spectrum	28	2	12	93.33	70.00	80.00	
	direct station selection	64	3	4	95.52	94.12	94.81	
EOT	*end of tape*	74	5	8	93.67	90.24	91.93	74.21
	end of text	66	58	17	53.23	79.52	63.77	*100.00*
	end of transmission	77	52	10	59.69	88.51	71.30	*100.00*
FMS	false memory syndrome	21	0	5	100.00	80.77	89.36	98.28
	financial management service	80	1	3	98.77	96.39	97.56	*98.28*
	flexible manufacturing system	71	0	3	100.00	95.95	97.93	*27.51*
	flight management system	65	1	7	98.48	90.28	94.20	
	foreign military sales	76	1	3	98.70	96.20	97.44	
	forms management system	8	0	7	100.00	53.33	69.57	
FOB	federal office building	44	2	7	95.65	86.27	90.72	91.76
	free on board	88	13	0	87.13	100.00	93.12	
	fresh off the boat	27	1	4	96.43	87.10	91.53	
FOC	faint object camera	64	0	7	100.00	90.14	94.81	94.41
	fiber optic communications	10	1	6	90.91	62.50	74.07	
	free of charge	74	8	0	90.24	100.00	94.87	

Table 7.1: Results of acronym disambiguation on the evaluation corpus (continued)

Acronym	Expansion	co	in	nf	pr	rec	F	acc
IFF	*identification friend or foe*	78	8	0	90.70	100.00	95.12	69.96
	image file format	59	40	23	59.60	71.95	65.19	*100.00*
	interchange file format	38	37	25	50.67	60.32	55.07	*100.00*
INS	immigration and naturalization service	79	7	5	91.86	94.05	92.94	94.02
	inertial navigation system	60	1	6	98.36	90.91	94.49	
	information network system	13	2	12	86.67	52.00	65.00	
	input string	75	2	1	97.40	98.68	98.04	
MAC	*mandatory access control*	63	0	3	100.00	95.45	97.67	96.12
	media access control	68	0	4	100.00	94.44	97.14	*97.67*
	military airlift command	86	7	0	92.47	100.00	96.09	*53.49*
	money access	28	2	6	93.33	82.35	87.50	
MCC	mennonite central committee	72	16	1	81.82	98.63	89.44	94.47
	metropolitan community church	61	8	7	88.41	89.71	89.05	
	mission control center	73	0	3	100.00	96.05	97.99	
MOD	magneto optical disk	63	1	4	98.44	94.03	96.18	94.64
	medical officer of the day	10	0	6	100.00	62.50	76.92	
	ministry of defence	85	9	0	90.43	100.00	94.97	
MPS	main propulsion system	26	4	4	86.67	86.67	86.67	95.03
	master production schedule	65	5	0	92.86	100.00	96.30	
	megabytes per second	26	3	7	89.66	78.79	83.87	
	mpoa server	52	0	1	100.00	98.11	99.05	
MTS	manitoba teachers society	27	1	5	96.43	84.38	90.00	79.48
	manitoba telephone system	22	4	13	84.62	62.86	72.13	*96.42*
	member of technical staff	39	0	6	100.00	86.76	92.86	*55.05*
	message telecommunications service•	80	34	7	70.18	91.95	79.60	
	message telephone service•	30	11	21	73.17	58.82	65.22	
	message transfer service◇	7	0	5	100.00	58.33	73.68	
	mobile telephone service•	16	8	18	66.67	47.06	55.17	
	multichannel television sound◇	8	0	3	100.00	72.73	84.21	
NSP	national ski patrol	54	3	0	94.74	100.00	97.30	94.05
	native signal processing	11	0	4	100.00	73.33	84.62	
	network services protocol	13	1	2	92.86	86.67	89.66	

Table 7.1: Results of acronym disambiguation on the evaluation corpus (continued)

Acronym	Expansion	co	in	nf	pr	rec	F	acc
PAL	passive activity loss	41	0	1	100.00	97.62	98.80	97.30
	phase alternate line	71	1	4	98.61	94.67	96.60	*98.92*
	programmable array logic	67	4	1	94.37	98.53	96.40	*77.30*
PID	pelvic inflammatory disease	75	0	2	100.00	97.40	98.68	97.95
	photoionization detector	82	0	1	100.00	98.80	99.39	*98.97*
	process identifier	72	2	3	97.30	96.00	96.64	*44.18*
	protocol identifier	53	0	4	100.00	92.98	96.36	
RIP	*raster image processor*	77	2	2	97.47	97.47	97.47	97.72
	rest in peace	70	3	2	95.89	97.22	96.55	*99.09*
	routing information protocol	67	0	1	100.00	98.53	99.26	*67.12*
RTP	rapid thermal processing	70	0	2	100.00	97.22	98.59	98.13
	real time protocol	62	0	2	100.00	96.88	98.41	
	research triangle park	75	3	3	96.15	96.15	96.15	
SAP	*second audio program*	64	1	2	98.46	96.97	97.71	93.09
	service access point	73	10	4	87.95	94.81	91.25	*100.00*
	service advertising protocol	62	5	12	92.54	83.78	87.94	*100.00*
SAR	school of american research	19	0	3	100.00	86.36	92.68	97.30
	search and rescue	63	2	7	96.92	90.00	93.33	*97.60*
	segmentation and reassembly	69	0	4	100.00	94.52	97.18	*48.95*
	shift arithmetic right	21	0	2	100.00	91.30	95.45	
	successive approximation register	69	0	1	100.00	98.57	99.28	
	synthetic aperture radar	69	3	6	95.83	92.00	93.88	
SDS	smart distributed system	52	0	5	100.00	91.23	95.41	95.52
	sodium dodecyl sulfate	80	6	0	93.02	100.00	96.39	
	students for a democratic society	70	0	5	100.00	93.33	96.55	
	switched data service	7	0	4	100.00	63.64	77.78	
SEM	*scanning electron microscope*	71	1	4	98.61	94.67	96.60	95.62
	standard electronic module	9	1	3	90.00	75.00	81.82	*100.00*
	standard error of the mean	73	4	0	94.81	100.00	97.33	*54.37*
SNL	sandia national laboratories	64	0	2	100.00	96.97	98.46	98.06
	saturday night live	65	4	2	94.20	97.01	95.59	
	school for new learning	21	0	1	100.00	95.45	97.67	

Table 7.1: Results of acronym disambiguation on the evaluation corpus (continued)

Acronym	Expansion	co	in	nf	pr	rec	F	acc
STC	set carry flag	16	0	2	100.00	88.89	94.12	99.43
	society for technical communication	70	0	1	100.00	98.59	99.29	
	supplemental type certificate	85	0	0	100.00	100.00	100.00	
TIFF	*tagged image file format*	71	56	10	55.91	87.65	68.27	64.17
	tagged information file format	18	0	14	100.00	56.25	72.00	*90.37*
	tag image file format	57	67	17	45.97	77.03	57.58	*82.89*
TSS	*task state segment*	33	1	0	97.06	100.00	98.51	94.90
	tethered satellite system	45	1	0	97.83	100.00	98.90	*100.00*
	time sharing system	14	0	6	100.00	70.00	82.35	*54.08*
	Totals	9,214	959	749	90.57	92.48	**91.52**	**92.58**
	Baseline without sense interaction	3,680	6,283	6,283	36.94	36.94	**36.94**	**36.94**
	Totals after sense interaction	8,836	285	629	96.88	93.35	**95.08**	***97.14***
	Baseline with sense interaction	4,419	4,702	4,702	48.45	48.45	**48.45**	***48.45***

Considering documents already converted into value vectors, the efficiency (running time) of the system is dependent upon the size in documents of the training set (usually $10^2 \leq R \leq 10^3$), the number of training features (usually $10^3 < f < 10^4$) and the number of senses (s). Asymptotically, training time: $T_t \in O(R^2 f^2)$, evaluation time: $T_e \in O(R^3 f^2)$, decision time for a new, unseen document: $T_d \in O(f)$, and selection time for a new document: $T_s \in O(fs)$.

The efficiency of the conversion from text into value vectors in feature space is linear in the size (number of words) of the document; an LALR(1) parser generated using Lex and Yacc, and hash-table dictionary creation and lookup are used. The components of the software are implemented in C++, Java and Perl. Evaluation is performed using multi-level (system, acronym, sense) *make* scripts.

Observed running times (measured on a machine with a 1.6GHz Pentium 4 processor, running RedHat Linux 7.3.) are less than 1s for selection per document, including text-to-value vector conversion, less than 5s for training per sense, and between 10–60min for evaluation per acronym (depending on the number of senses and documents). Running the system on all vectors in the evaluation corpus took 25h15min.

The main bottleneck of the system is the *ps2ascii* utility, both in terms of efficiency and quality of the output. Higher quality software packages with similar functionality are available under commercial licensing arrangements.

The Google API has a built-in limitation of 1,000 queries per day, which imposes serious, but finite restrictions to the speed of acquisition of training data. This limitation can be overcome given direct access to search engine index databases, or alternative licensing arrangements.

The constants G and L are based on properties of the search engine used: total number of indexed documents, respectively average number of lists of acronyms returned from a query. They can be adjusted based on the choice of search engine, or with changes in the coverage of the search engine. The values of other constants are chosen after experimentation on the development corpus.

The value of the constant w^+ is justified by the desired bias of the system: in favor of false positive errors or in favor of false negative errors. Its value can be adjusted based on specific requirements. The value $\overline{\pi} = 3$ is chosen to separate as much as possible between situations of true and false sense interaction. Manipulation of this constant will affect the performance/baseline trade-off when considering grouping of interacting senses.

The values of the constants $\gamma_A = 100$ and $\gamma_t = 10000$ are chosen to allow for whole lists of acronyms returned by the search engine and respectively, very long documents to be avoided during training, without restricting significantly the size of the training set. These values can be adjusted, resulting in manipulation of those parameters. During development, we have found that choices within value ranges $20 \leq \gamma_A \leq 500$ and $5000 \leq \gamma_t \leq 50000$ result in insignificant changes to the end results.

The constants α, β are parameters in the calculation of "distance" between a given acronym and a given word occurrence. They govern the weights associated with the presence of an term within a given document and, respectively, the maximum-sensitivity window around an acronym occurrence. During development, we have found that choices within value ranges $0.1 \leq \alpha \leq 1$ and $0.5 \leq \beta \leq 2$ do not affect the end result significantly. Manipulation of the values of these constants will be needed for increased performance when processing specific categories of text.

7.6 Conclusions

A general solution is presented to the problem of acronym sense disambiguation in text. The system described uses support vector machines (SVM), a machine learning technique that has been successfully used in numerous other natural language processing problems, including acronym acquisition.

The experimental results, for acronyms longer than three letters in the English language yield performance superior to that of general-purpose WSD systems.

The smallest performance figures are consistently reported for senses with a small number of documents used during training. Larger training sets are expected to improve performance. A small number of available training examples is also a symptom of the low frequency of a given acronym sense, leading to low performance only for acronyms that are *less frequently used* in the covered domain (indexed by search engines).

The automatic acquisition of training data, starting from dictionaries of acronyms and using documents available on the Internet, returned from search engine queries, is a low cost alternative to the major expense of building WSD training corpora. Automatic leave-one-out cross validation on training data reduces the cost and the bias involved in building evaluation corpora.

The main limitation of the system comes from the coverage of the acronym dictionary. The results of evaluation do not take into account senses not encountered before, for which example documents are, obviously, not downloaded or used during training. In other words, the system cannot disambiguate occurrences of acronyms with senses unencountered during training. It follows that an acronym dictionary with adequate coverage is a prerequisite of an usable acronym disambiguation system. In special cases, where acronyms that do not appear in the database, but their expansions show up in the text under analysis, acronym sense disambiguation can be supplemented with acronym acquisition (future work).

The performance of the system on given acronyms and senses, as calculated during evaluation, can be used in runtime conditions to qualify the confidence in selection of a given sense on a new document. For example, if the accuracy of selection for a given acronym is high, but the precision in choosing a particular sense is low, a "low confidence" warning will be reported to a user, should that sense be returned by the system.

Interaction between senses of the same acronym that are either related or overlapping can be automatically evaluated, even in the absence of training data, through the calculation of perplexity of cooccurrence of the expansions for search engine hit counts. Grouping of senses using such relatedness measures increase the reported performance of the system, considering that disambiguation of interacting senses is not required. This increase in performance occurs at the expense of an increase in baseline performance. This mechanism cannot differentiate (future work) between cases of false polysemy, such as "Computer Aided Instruction" or "Computer Assisted Instruction" for 'CAI', and interaction between true senses of the same acronym form, such as "Canadian Transport

Commission" or "Centralized Traffic Control" for 'CTC'. The system has an ability to identify, without using training data, situations where it is expected to do poorly. It is, consequently, appropriate to use sense interaction information only in conjunction with acronym forms for which evaluation yields lower performance (e.g. <90%).

The system can be expanded incrementally to incorporate new senses without a need for all training data to be regenerated. Its high level of automation, performance and modularity make it adequate for general-purpose acronym sense disambiguation on the web, currently under development by the author.

Chapter 8

General Conclusions and Future Work

This work originated from a recognition of the need for a comprehensive view of acronyms from a multilingual perspective, which will result in modular, efficient implementations of automatic acronym systems.

A comprehensive taxonomy which includes acronyms as well as related and overlapping phenomena is presented. Tests which differentiate between ambiguous categories are proposed. Such a taxonomy was not available prior to this work, and its main purpose is to provide consistent terminology for future systems dealing directly or indirectly with acronyms.

A novel, universal explanatory theory for acronyms is proposed, which attempts to account for all acronyms in all languages. The theory rests on a number of hypotheses, which can be tested linguistically, and is formulated as a set of fifteen universal hierarchical violable rules, with different activations and orderings for different languages. The theory is developed based on examples in fifteen languages (English, Spanish, French, German, Finnish, Italian, Hungarian, Romanian, Russian, Bulgarian, Hebrew, Arabic, Farsi, Chinese and Japanese), using six different writing systems (Latin, Cyrillic, Hebrew, Arabic Chinese/Kanji and Japanese Kana) and is automatically tested on lists of acronyms in eleven languages (Russian, Spanish, Danish, German, English, French, Italian, Dutch, Portuguese, Finnish, and Swedish). The theory accounts for all studied acronyms.

A multilingual, comprehensive theory of acronym formation was not available prior to this work. Its significance lies in providing exhaustive coverage for the underlying phenomena of acronym formation, a preliminary step for any acronym-related algorithms with wide, multilingual scope.

The theory results in an implementation of an efficient dynamic programming algorithm, based on a hypothesis of optimal substructure of acronym formation regularities. The high accuracy (over 99% for ten languages and 98.58% for Russian) of this algorithm supports the hypothesis of optimal

substructure, which provides a strong theoretical framework for a set of natural language phenomena (acronyms). The fit of the framework is surprisingly adequate, compared to the fit of linguistic models of other natural language phenomena.

Due to the violable nature of rules in the presented theory, possible future contradictions generated by new observations are expected to be resolved through reordering of the rules for specific languages and through the addition of new, possibly more specialized rules. It is also expected that any theory of acronyms, no matter how comprehensive, will necessarily be incomplete, since it must follow the regularities of very complex human competence, which will likely (and hopefully) escape perfect modeling by computer software in the foreseeable future.

The main problems related to acronyms are automatic acronym acquisition and acronym disambiguation. A novel, modular strategy for approaching these problems is presented.

Acronym acquisition is based on the identification in text of acronym definitions, which are a form of anaphora or cataphora. It is submitted here that acronym acquisition is a special case of anaphora or cataphora resolution and that acronym disambiguation is a special case of the general problem of word sense disambiguation.

Since anaphora resolution and word sense disambiguation are two of the most difficult and elusive open problems in natural language processing, their formulation in restrictive, rather than generic cases, is one of the most promising courses for advancement. Acronym acquisition and disambiguation are two such restrictive formulations, for which efficient and accurate solutions are introduced in this thesis.

Two classes of acronym acquisition systems are presented: one using linguistic heuristics in a greedy algorithm, implemented in a logic programming formulation, the other a modular approach, based on machine learning using the parametrized output of an acronym-expansion matching system. Both systems achieve good performance for specific and generic text in the English language. Although corpora used in the evaluation of systems in the reviewed literature are not available, the performance reported for the systems presented here is superior to that of any comparables. Since both acronym acquisition systems are based on the universal theory of acronyms, whose fit has been evaluated on a number of different languages, it is expected that these systems will be applicable directly, without modifications in principle, to other languages and domains. Such an effort, with promising results has already been undertaken by the author for newspaper text in the German language and for the text of medical abstracts, although the results are not published.

Acronym sense disambiguation for general text in the English language is executed using an unsupervised machine learning approach to a special case of word sense disambiguation, where

the senses of acronyms are taken to be the expansions. The approach achieves good accuracy for acronyms, and automatically acquires training examples, starting from a dictionary of acronym definitions. Unsupervised evaluation of the system is also conducted, on automatically downloaded test data.

Acronym disambiguation for general text was not studied prior to this work. The performance of the system, on general text is superior to acronym disambiguation attempted in the medical domain.

The acquisition of a large database of acronyms, starting from web documents, is currently in progress, using technologies developed as part of this work. The evaluation of the acquisition and disambiguation methodologies in other languages and domains is left as future work.

I submit that the presented acronym acquisition methodologies are applicable to a wider class of noun phrase anaphora resolution problems, as solutions with a *named entity recognition* focus, following generalization of the modular approach based on matching of entities and machine learning of definition patterns. Quantitative tests of this hypothesis are left as future work.

I submit that the methodology of the acronym sense disambiguation system is applicable to more general word sense disambiguation (WSD) problems, especially the generalization of automatic acquisition of training and evaluation data, which represents one of the major cost items of WSD systems. Testing this hypothesis is left as future work.

Appendix A

Acronyms ∩ (NLP ∪ CL)

This appendix covers a number of tasks and areas of natural language processing (NLP) and computational linguistics (CL) that are applicable to acronym acquisition and disambiguation in different languages and domains.

Methodologies and algorithms in the areas of phonology (syllabification), morphology (derivational and compound), syntax (partial parsing), discourse (anaphora) and semantics (word sense disambiguation) are investigated. A number of computational linguistic formalisms are presented, including Optimality Theory (OT), File Change Semantics (FCS), and Centering.

Common elements and differences are stressed out, algorithmic and computational aspects are discussed.

A.1 Motivation

This thesis defines and follows a systematic approach to acronym acquisition and disambiguation, illustrated in Figure 1.1 and discussed in section 1.2.

The acronym identification task must deal with ambiguities such as whether HELP in a specific context is the noun/verb "help", capitalized for emphasis, or an expansion of "Health, Education, Labor and Pensions" (a US senate committee).

Such ambiguities are resolved using part-of-speech tagging, lexicon lookup and word sense disambiguation techniques.

I have proposed in section 2.4 an explicative theory for acronyms, which is based on preferential matches at word, morpheme, phoneme and letter boundaries.

The automatic identification of morpheme boundaries relies on derivational morphology, as in

the example: HTTP, and acronym for "Hypertext Transfer Protocol", where hypertext is, arguably, a derived form of the noun "text", with the prefix "hyper". Compositional morphology is also be an important factor in specific domains or languages, as in the example: DNA, an acronym for "deoxyribonucleic acid", where deoxyribonucleic is a compound word de(derivational prefix)- oxy- ribo- nucleic.

The automatic identification of phoneme boundaries relies upon phonology and syllabification resources, as in the example: MKTG, an acronym for marketing, syllabified (mar)(ke)(ting).

The expansions of acronyms are generally stand-alone noun phrases, that can be parsed using complete or shallow syntactic parsing methodologies.

Expansion identification relies on the automatic identification of acronym definition patterns, which can be modeled by a limited number of regular expressions (and consequently automata), or by more involved discourse-level mechanisms such as anaphora; in the following example [64], the expansion of the acronym NIR (Networked Information Retrieval) is introduced:

> The purpose of this report is to increase the awareness of Networked Information Retrieval by bringing together in one place information about the various networked information retrieval tools, their developers, interested organisations, and other activities that relate to the production, dissemination, and support of *NIR* tools.

More complex definition patterns require both semantic and complete syntactic analysis, as in the following example, which provides an expansion for the acronym BMA (Broadcast Multiple Access) based on an assumption inferred from the definition of the acronym NBMA (Non Broadcast Multiple Access) from [64]:

> (...) use within Non Broadcast Multiple Access (NBMA) networks; although, a somewhat straight forward usage is applicable to *BMA* networks.

Occurrences of acronyms undefined in context need to be disambiguated using, for example, word sense disambiguation techniques.

A.2 Phonology

Phonology is "the area of linguistics that describes the systematic ways sounds are differently realized in different environments, and how this system of sounds is related to the rest of the grammar", [68, pg. 92].

Prosodic categories, which (including their relationships) form the object of study of phonology, [92] can be represented as a length-based hierarchy, containing (bottom-up), the mora (μ), the syllable (σ), the metrical foot (F) and the prosodic word (PrWd).

For all languages there is a relationship between the surface representation (written form) and the underlying representation (pronunciation). In languages such as English, this relationship is not entirely deterministic; exceptions from the rule are common.

The main applications of phonological theories are TTS (text-to-speech) generation, and speech recognition, but areas such as syllabification also have written text applications.

A.2.1 Optimality Theory

Optimality theory (OT) is a general linguistic theory, claimed, [6] to be "THE linguistic theory of the 1990s". OT has its roots in Chomsky's "principles and parameters" theory, [27], reprinted in [26].

OT was introduced by [115] and has been the dominant approach in computational phonology up to around the year 2000. The Rutgers Optimality Archive[1] is an online repository of electronic versions of a significant number of published and unpublished papers on Optimality Theory.

OT's notable successes have so far been limited to phonology, even if approaches to syntax (e.g. [17], in conjunction with the LFG formalism) or semantics [58], [12] are attempted.

OT is trying to account for a wealth of linguistic phenomena through a three-tier mechanism for mapping input (or a surface form) to output (an internal, or underlying form):

- GEN : a generator function that relates input to a set of candidate representations
- CON : a partially ordered universal set of violable constraints. As the set of constraints is applicable to every language, constraints have specific orderings for individual languages.
- EVAL : a mechanism that selects candidate representations that best satisfy the ranked constraints.

At the core of each OT solution is a list of constraints (CON), which is considered to be universal for each given domain (invariant for all natural languages for that domain, e.g. phonology), an instantiation of Chomsky's thesis of Universal Grammar (or "principles"). The ordering of the constraints is considered to be the main mechanism (the *only* mechanism in most cases) to account

[1] On the web http://roa.rutgers.edu (February 2004).

for language variability (or specific language "parameters"). OT has found fertile ground in phonology perhaps because "constraint ranking was silently present in the phonological literature for many years", [12].

The superiority of violable constraint-based formalisms in phonology vs. (constructive) rule-based formalisms is argued by example in works such as [51]. The argument is based on the graceful degradation of the performance of constraints-based systems in the face of so-far unseen observations. By contrast, constructive rule-based systems (even based on violable, non-strict rules) need to account for unseen data, or fail in the presence of data not seen before. In summary, [51]:

> Under a rule-based account, the only way to predict a typology is to change rules into constraints.

Unfortunately, following this same argument, in order not to sacrifice overall coverage, the GEN component of OT systems must have complete coverage over the space of possible mappings of input to output, which can lead, in many cases, to a computational explosion. For example, in the syllabification of a word, in order to guarantee complete coverage, GEN needs to list all the possible ways of splitting the word in groups of letters, which is exponential ($\Theta(2^n)$) in the length of the word (n).

It follows that for computationally feasible OT algorithms, a tighter integration of the GEN and EVAL modules must be present, rather than a loose interface where all forms generated by GEN are fed into EVAL.

Going further, any OT system must rely on the successful execution of the following phases:

- (U) Identification of universal constraints based on knowledge of many (all ?) natural languages.
- (R) Ranking of constraints for a specific language, in the presence of experimental data, attested forms, cf. [35].
- (G) Generation of an underlying form from a surface form.

Phase (U) above is based on linguistic discovery methodologies, which identify classes of utterances in various languages which are unacceptable for native speakers, and observations of phenomena such as expletive infixation or syncope. The task is compounded significantly by the strong

requirement for the set of constraints to be universal (cross-linguistic). Automatic methods of identification of such constraints is highly problematic, if at all possible, and is challenged by the heterogenous nature of most proposed constraint systems, and their violable nature.

The differential ranking of pairs of constraints is, again, based on observations of ill-formed utterances and other linguistic investigative mechanisms. Under some restrictions on the nature of constraints (e.g. constrains modeled using finite state automata, or regular expressions), efficient algorithms, based on "topological sort" can be imagined, [35].

A much more complex problem, although not falling directly within Optimality Theory is the complete ranking (R) of a given set of constraints, in the presence of experimental data, proved in [35] to be at least as complex as the discovery of Hamiltonian cycles in graphs, a problem known to be NP-complete. Specific instantiations of this problem are shown to be exponential.

Even the generation (G) of an underlying form, given a specific surface form and an ordered set of constraints is shown to be hard (at least NP-complete) in most cases.

A credible argument to indicate the inadequate computational complexity (intractability) of OT solutions is provided [35]. Its conclusion (fragment following) has probably provided a chilling effect on subsequent OT work:

> Hence all OT generation and learning algorithms should be suspect. Either they oversimplify their problem, or they sometimes fail, or they take worse than polynomial time on some class of inputs. (Or they demonstrate P = NP!)

Indeed, the dominance of OT papers at SIGPHON in the late nineties is no longer present at SIGPHON-2002.

Results of optimality theory are best represented and supported by software in the area of phonology. [50] and [52] are implementations of syllabification systems for the English and French languages that I have evaluated on an architecture based on a Pentium 6, 1.6 GHz processor, running the Linux operating system. Precision, recall or other performance measures are not presented in the original papers.

These implementations are reliably correct (evaluation is not provided) for shorter words, they time out for words exceeding 20 letters and provide incorrect results for some morphologically complex terms, e.g. the Hammond syllabifier [50] syllabifies incorrectly cardioplegic as *car-di-op-legic, rather than car-di-o-ple-gic, (following morpheme boundaries cardio-plegic) and also syllabifies incorrectly aethoxysclerol as *a-et-hoxysc-le-rol, not accounting, for instance, for the vowel 'y' in the morpheme "oxy".

These inaccuracies are indicative of an incomplete constraint set, as in the case of the vowel 'y', and also indicate that additional (morphological) constraints need to be added to the system and that phonological variations within specific domains need to be accounted for, such as the abundance of "neo-classical" (originally Latin and ancient Greek) terms in the English medical terminology.

A.2.2 Other Approaches to Syllabification

[14] reviews a number of methodologies for learning pronunciation, including syllabification, from corpora. Connectionist, decision tree-based and inductive learning are compared on examples, as well as on entries from the lexical database CELEX, [18].

More recently, [102] uses probabilistic context-free grammars (PCFG) to model the word and syllable structure of German and English. The probabilities of productions in large grammars for German (2,394 context free rules) and English (4,418 rules, semi-automatically derived from the German grammar) are trained on large corpora in a supervised mode. The system achieves an accuracy of 96.88% on German. Results are not reported for English syllabification.

A.2.3 Conclusions

Optimality Theory (OT) is perhaps the most influential recent descriptive theory of phonology. It provides accounts for phenomena such as syllabification, stress and accent, through a constraint-based formalism motivated by Chomsky's theory of principles and parameters.

It has been proven [35] that OT systems are prohibitively complex computationally, at least NP-complete, both from the point of view of design: identifying and ordering constraints given evidence, as well as from the point of view of implementation: providing analyses of utterances. OT implementations for syllabification work reasonably well on short words.

Other approaches to syllabification use statistical tools, such as PCFGs, and machine learning. Even relatively simple tasks, such as syllabification, prove to be challenging in languages such as English, with highly irregular graphemic systems.

Syllabification is a prerequisite of the support of rule 2.3 in the universal acronym formation theory presented in section 2.5. The systems presented in chapters 3, 4 and 5 rely on syllabification of the expansion or expansion candidates and use the Hammond [50] OT syllabifier. Although it is not clear that acronym formation follows syllable boundaries, syllabification is used as a readily available, less precise substitute for the identification of morphemes in morphological compound terms.

A.3 Morphology

Morphology is an area of linguistics that focuses on word-centered variations. The *morpheme* is the basic semantic unit of language and the minimal unit of linguistic form with meaning. A morpheme can be realized as:

- *Stem*: the mandatory morphological component of a word, usually occurs at the central part of a word. Also known as *root* or *base*.
- *Prefix*: occurs before a stem, usually at the beginning of words.
- *Suffix*: occurs after a stem, usually at the end of words.
- *Infix*: occurs within words and acts as a modifier of the sense of the stem. Infix only occurs in certain languages.

Prefixes and suffixes are also called *affixes*. Prefixes, suffixes and infixes do not have meaning on their own and act as modifiers to the meaning set by the stem or stems. They are called *bound morphemes*. Stems can occur as individual words, called *free morphemes*.

Most languages produce morphological forms that can be described through the following regular expression, where '+' stands for "occurs at least once, possibly repeated", and '*' stands for "occurs or not, possibly repeated":

$$\{Prefix\} * \{\{Stem\} + \{Suffix\}*\}+ \tag{A.1}$$

Many times changes of form in the stem occur through the addition of affixes, through phenomena that are principally justified by phonology, such as the addition of an extra 'e' in the plural of the noun "process" + 's' = "process*es*."

Productivity in morphology occurs through the following three principal phenomena:

- *Inflection*, where a word changes form, in so-called *paradigms*, without changing word class. Such changes consist of changes to number, tense, case, etc. For example, the third person singular inflection of the verb "go": "goes", is realized by adding the suffix 's'.
- *Derivation*, where a word changes word class, by a process such as the addition or substitution of an affix. For example, many adjectives in English can form related sense adverbs, through the addition of the suffix -ly, such as "sad" + "ly" = "sad*ly*".

- *Compounding*, where multiple stems are combined to form a word with combined, or sometimes new, meaning. For example, the formation of the compound noun "surfboard" from the components "surf" and "board."

Languages such as German, Finnish or Turkish have rich compound morphologies. For example, in Finnish, [31] the word "kahvinjuojallekin", with translation "also for [the] coffee drinker" is a compound formed from (English morpheme translations in square brackets):

$$\text{kahvi [coffee]} + \text{n [of]} + \text{juo [drink]} + \text{ja [-er]} + \text{lle [for]} + \text{kin [also]} \tag{A.2}$$

While the mainstream of the English language is less productive in this respect, there are specialized areas where compounds are very common. Such is the case for medical and biochemical terminology, where stems/roots of Ancient Greek and Latin etymology are combined to create so-called *neo-classical compounds*. For example, the word "deoxyribonucleic" is created through concatenation (affix labels in square brackets):

$$\textit{de} \text{ [prefix]} + \text{oxy} + \text{ribo} + \text{nucle} + \textit{ic} \text{ [suffix]} \tag{A.3}$$

Since morphemes in the examples (A.2) and (A.3) above can be represented in languages without compound morphology as separate words, a syntactic structure, presented in Figure A.1 can be imagined on top of the unidimensional structure of morpheme decomposition.

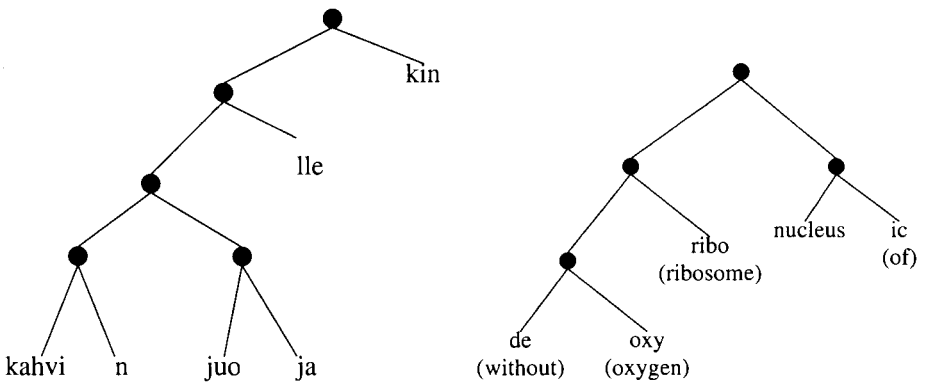
The number of legal compound terms that can be constructed with a finite number of roots can be very high, through exponential explosion, which makes it impractical to list all compounds, even if just the ones manifested, in dictionaries or specialized lexica. For example, the UMLS[2] corpus, [105] contains 2,279 neoclassical roots, in a list that is not exhaustive. Using these roots, a number of compound words in excess of 1.18 billion ($2,279 + 2,279^2 + 2,279^3$) can be formed with 3 or less roots.

Most work in computational morphology is focused on inflectional and derivational morphology, given its direct applicability to the English language.

A.3.1 Two-level morphology

One of the first computationally feasible models of inflectional and derivational morphology is Koskenniemi's 1983 two-level morphology, presented in [75].

[2]On the web http://www.nlm.nih.gov/research/umls (February 2004).



(a) Syntactic parse tree of the Finnish compound noun "kahvinjuojallekin", with decomposition shown in (A.2)

(b) Syntactic parse tree of the neoclassical compound "deoxyribonucleic", with decomposition shown in (A.3)

Figure A.1: Syntactic parse trees associated with morphological compounds (Figure by author based on work by Creutz and Lagus, 2002)

Two-level morphology is based on grammar productions between a phonemic or graphemic surface representation, and a deep, morphophonemic representation. The highest level of complexity of these production is "context-dependent", and mechanisms for "compilation" into finite-state automata are also provided.

Processing using the resulting FSA is linear in the size of the input, but is formally proven, [8], to be at least NP-complete in the size of the grammar. [76] offer considerations following which rules extracted from the morphologies of natural languages will not lead to a computational explosion, by showing that the number of non-regular such rules in a grammar is usually very small. Their discussion is related to *vowel harmonies* and other potential sources of compounding in various natural languages, the number of which dictates a potential exponential explosion in the corresponding FSA grammar. The authors claim that there are computationally feasible bounds on the size of such grammars, since languages with more than two vowel harmonies are not known.

Two-level morphology is the basis of inflectional and derivational morphology analyzers, such as PC-KIMMO[3], implementations with good performance for English and Finnish. As noted in [128], however, the main drawback is the difficulty to achieve a reasonably wide coverage of complex

[3]On the web http://www.sil.org/pckimmo (February 2004).

phenomena.

KIMMO-like systems are also used for solving compound morphology problems, and can be enhanced with part-of speech information to a morphological parser, [3].

A.3.2 Partial morphological analysis: stemming

A simpler problem than complete morphological analysis is the identification of stems from inflected or derived forms. Heuristic systems for the English language, such as the Lovins stemmer[4], [89] are still highly effective as modules in areas such as Information Retrieval (IR).

Without claiming to support a strong theoretic formalism, such as two-level morphology, the *Porter stemmer*[5] algorithm, [114], originally published in 1980, is an aggressive stemming algorithm. Similarly with the Lovins stemmer, it uses a number of heuristics and is used as a module in bigger special purpose systems, which are focused on processing speed.

[146] report problems in the Porter stemmer such as over-conflation: e.g. the words "policy" and "police" are assigned the same stem, and over-expansion: e.g. the word "iteration" is expanded into the inexistent root "iter" and the suffix "ation". They improve the accuracy of the Porter stemmer in these areas by using word co-occurrence information from text to gather evidence of morphological relatedness and reduce over-conflation and over-expansion effects.

The justification for partial morphological analysis, stemming, comes from improvements in other NLP tasks, such as IR, in conditions when stemming is part of the pre-processing. [62] is a comprehensive review of stemming algorithms from this perspective, which concludes that stemming marginally improves the performance of IR systems, by 1-3%, with marginal differences on average between the different stemming algorithms evaluated.

In specific document repositories and queries, this impact is much more significant, [78] reports improvements of up to 35%.

A.3.3 Statistical methods

Since the generation by humans of morphological rules for new languages in formalisms such as two-level morphology can be expensive and time-consuming, a more recent trend is based on the automated learning of morphological regularities based on raw corpora, in an unsupervised or mildly supervised manner.

---

[4]On the web http://snowball.tartarus.org/lovins/stemmer.html (February 2004).

[5]On the web http://www.tartarus.org/~martin/PorterStemmer (February 2004).

[44] presents an approach that automatically learns inflectional and derivational morphology, based on the information-theoretic notion of *Minimum Description Length* (MDL), and a number of heuristics. Using a *model*, or codebook, that associates codes to morpheme candidates, the compressed size of a corpus is calculated as the size after replacing all occurrences of morpheme candidates with codes from the codebook. The *description length* of the corpus is the sum of the compressed size of the corpus and the size of the codebook.

The algorithm minimizes the description length of the corpus by iteratively proposing decompositions and measuring their impact on the description length of the corpus. Decompositions which reduce the description length are accepted as evidence in favor of morphemes. Goldsmith achieves approximately 86% precision, 90.4% recall for English and 87% precision, 89% recall for French.

[122] approach inflectional morphology by building hierarchies of possible affixes and using semantic information, Latent Semantic Analysis (LSA) for validating stems. Their system achieves similar performance, 84.3% precision and 90% recall, indicating that semantic analysis, even if incomplete, provides cues for morphological relatedness of terms to be analyzed.

A.3.4 Compound morphology systems

Much less work has been done in the area of compound morphology. Although models such as two-level morphology are also useful here, lexical knowledge is essential, whether extracted from machine-readable dictionaries or learned from corpora.

Some recent developments include [156] and [157], who present a methodology for the extraction of morphological knowledge from domain-specific thesauri (such as in the medical domain). Their method relies on the existence in the thesaurus synonym series for a given term of words with common morphemic constituents. The resulting system achieves precision of 97.3% for French, 91.9±1.5% for English and 99.6% for Russian, and a recall of 91.2% of inflection variations and 79.2% of the derivation variations for English.

[31] uses *minimum description length* and *maximum likelihood* (ML) to infer morphological compounds from a corpus, achieving correct complete decomposition for 49.6% of entries and correct, but incomplete decomposition for 29.7% of entries, on a random sample of Finnish compounds. As evaluated, their system outperforms the system in [44], following similar considerations by the author: "more work remains to be done to identify compounds in general".

A morphology system for German, MORPHIX[6], is also worth mentioning, even if no performance figures are readily available. MORPHIX identifies compounds through the recursive traversal of TRIE storage structures and the use of two-level rules.

A.3.5 Conclusions

Most work to date in computational morphology is related to the resolution of inflectional and derivational morphology. Partial analysis heuristic-based systems, such as the Lovins and Porter stemmers are useful and very efficient components in areas such as Information Retrieval.

Two-level morphology, a computational formalism based on finite-state automata is widely used. Even if formally proven to result in NP-complete representations, evidence suggests that the morphology of natural languages can be adequately described. The main drawback is represented by the high cost of designing morphological grammars.

Recent work focuses on unsupervised learning of inflexional and derivational morphology from corpora for new languages, with encouraging level of success. Information-theoretic and statistical measures such as minimum description length and maximum likelihood are used, as well as weak semantic approaches, such as Latent Semantic Indexing.

Compound morphology has been, comparatively, less explored. Recent approaches use specialized thesauri, such as from the medical domain, to infer morphological compounds, with good level of success. Promising results, even if more modest, are achieved in languages with productive compound morphologies, such as Finnish or German.

Acronyms match terms in expansions at intra-word morpheme boundaries, as stipulated by rule 2.2 and as quantitatively shown in chapter 6. All figures below refer to results reported in Table 6.5), for the corpora used in chapter 6. For languages with reduced morphological productivity, such as Latin languages and English (0.05–0.07 expected intra-word morphemes per word) acronyms rarely match intra-word morphemes (0.03–0.05 expected number of acronyms matching intra-word morphemes per total number of acronyms). For languages with rich compound morphology, such as Germanic languages, Finnish and Russian (0.13 for Russian, 0.32–0.51 expected intra-word morphemes per of word), acronym letters match a significant number of intra-word morphemes (0.13 for Russian, 0.42–0.89 expected number of acronyms matching intra-word morphemes per total number of acronyms). Similar considerations apply to domains with rich compound morphology, such as medicine or chemistry, where so called *neo-classical* compounds abound.

[6]On the web http://www.dfki.de/ neumann/morphix/morphix.html (February 2004).

The resolution of inflectional and compound morphology for words in expansions or expansion candidates is a prerequisite for acronym acquisition systems. In chapters 3, 4 and 5, lists of inflectional prefixes and syllabification are used to approximate the productivity of inflectional morphology in English. Entries such as the acronym 'DNA' (section 5.4) are missed by the hybrid system presented in chapter 5 due to the lack of compositional morphology resources, and incorrect syllabification of the morphological compound "deoxyribonucleic".

The availability of compositional morphology resources is a prerequisite of acronym acquisition in some languages (such as Germanic, Finno-Ungric, Slavic, etc.) and in some domains — such as medical text — with rich compound morphology.

A.4 Partial Parsing

Full syntactic analysis is generally considered necessary for full semantic analysis of natural language text, as well as for solutions to other problems as ambitious as that.

Since full syntactic analysis can, many times, be too resource intensive or imprecise, many applications need to rely on smaller scale solutions, which attempt to solve "easier" problems, but more precisely and more efficiently. Partial syntactic analysis, also named *partial parsing* or *shallow parsing*, serves this purpose.

A.4.1 Parsing by chunking

Psycholinguistic evidence is shown [43] that humans process language utterances in so-called *performance structures* (word clusters) and propose a model – *ϕ-phrases*: word sequences ending on syntactic heads that are content words, with some exceptions – that predicts empirical observations of word clustering.

Their observations are based on tree-like structures built upon pauses between elements of an utterance. In an influential work Abney [1] calls words clusters "chunks" and defines them in terms of "major heads", which account for better prediction of experimental data than in [43]. Abney's "major heads" are defined as:

> (...) any content word that does not appear between a function word f and the content word f selects, or a pronoun selected by a preposition.

Abney proposes a "chunking" syntactic analyzer that operates in two stages:

- *Chunker*: finds chunks in the input
- *Attacher*: finds higher-level relationships between chunks

The chunker used by Abney is a nondeterministic LR-parser (shift-reduce parser) based on a best-first search through a space of possible parses using a "score" computed on the basis of lexical frequencies, category preference, LR-conflict resolution and agreement.

Following Abney's work, an active field of "shallow parsing" or "partial parsing" is trying to approach a – supposedly – more manageable subset of the problem of full parsing of natural language. The idea is to isolate stand-alone chunks of utterances and combine them together into bigger chunks.

A.4.2 The current state of the art

Currently, shallow parsing is an important building block for natural language technologies and systems, with wide applicability in areas such as information extraction and text summarization.

There are a multitude of shallow parsing systems, based on a wide variety of technologies. The CoNLL conference's "shared tasks" for 2000 and 2002 have been related to shallow parsing, for two different levels where this problem can be raised:

- identification of non-overlapping phrases in text; shared task of CoNLL-2000, [135] compares the performance of 11 submitted systems on a phrase identification task
- identification of clauses in text; shared task of CoNLL-2001, [136] compares the performance of 6 systems on task of identifying clause boundaries within sentences

Both shared tasks use syntactically annotated evaluation data from the Penn Treebank corpus, [91].

The 11 systems submitted to CoNLL 2000 can be classified as follows:

- three rule-based systems; lowest performance on average: $F_{\beta=1}^{\text{avg.}} = 89.16 \pm 3.14$
- four statistical systems; higher performance on average: $F_{\beta=1}^{\text{avg.}} = 91.54 \pm .94$
- one memory-based system: $F_{\beta=1} = 91.54$
- three combined systems; highest performance on average: $F_{\beta=1}^{\text{avg.}} = 93.1 \pm .53$

Figure A.2 illustrates the distribution of the results (precision and recall) for the systems that have been submitted to CoNLL 2000 and Table A.1 presents a consolidation of the results, including references.

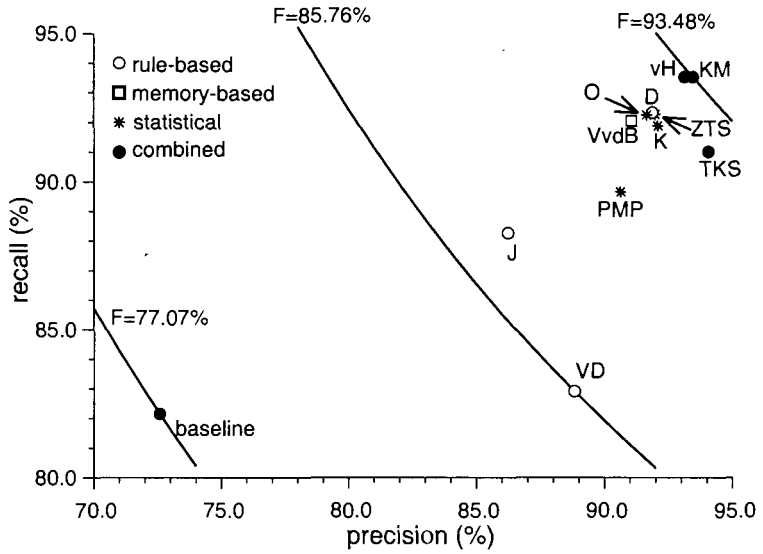


Figure A.2: Distribution of precision and recall on test data for the "chunking" shared task at CoNLL-2000. The curves of equal $F_{\beta=1}$ are indicated for the best and worst performing systems, and for the baseline (obtained by selecting the most frequent chunk tag for each part-of-speech tag). Systems are labeled with the identifiers in Table A.1 (Figure by author, based on results by Tjong and Déjean, 2001)

Most rule-based systems have a poor performance, with the exception of [34], which induces simple rules, which are shallow context dependent productions, through a theory refinement process, based on contextualization (learning of contexts) and lexicalization (learning of use patterns of specific words). The resulting set of rules is ordered in classes, based on the nature of the phrase category recognized, and used this way during evaluation.

Statistical based systems have reasonably high and close to each other performances, the highest achieved by [155]. Their system uses a standard HMM tagger, trained with the forward-backward algorithm and used, during evaluation, with the Viterbi algorithm. The HMM tagger is enhanced with some shallow contextual information. The system learns from training data, and errors are used

system	type	id.	precision	recall	$F_{\beta=1}$
[80]	combined	KM	93.45%	93.51%	93.48%
[138]	combined	vH	93.13%	93.51%	93.32%
[134]	combined	TKS	94.04%	91.00%	92.50%
[155]	statistical	ZTS	91.99%	92.25%	92.12%
[34]	rule	D	91.87%	92.31%	92.09%
[74]	statistical	K	92.08%	91.86%	91.97%
[111]	statistical	O	91.65%	92.23%	91.94%
[141]	memory	VvdB	91.05%	92.03%	91.54%
[113]	statistical	PMP	90.63%	89.65%	90.14%
[67]	rule	J	86.24%	88.25%	87.23%
[142]	rule	VD	88.82%	82.91%	85.76%
baseline			72.58%	82.14%	77.06%

Table A.1: Performance of eleven participating systems at CoNLL-2000. The field 'type' indicates "memory" for memory-based systems, and "rule" for rule-based systems (Table by author, based on results by Tjong and Déjean, 2001)

in an additional "error-based learning" step, followed by memory-based learning, which is executed during the testing phase.

Combined systems use both symbolic approaches and statistical methods and outperform all other systems, when comparing the $F_{\beta=1}$ measure. The system with the highest performance is [80], which uses *Support Vector Machines* (SVM) [29], [140] to learn a classification of sequences of tags 'inside', 'outside' and 'at the border' of chunks. The twenty possible combinations of such tags are used as features, with observations in a multi-dimensional vector space. SVMs, which currently represent a widely successful machine learning algorithm, prove to also provide impressive accuracy for chunking.

With the common advancement of full parsing and partial parsing algorithms, as the gap between current and adequate solutions is being reduced, so is the gap between performance of full and partial parsing. [85], compare a full parser with a shallow parser, both learning a common task: identification of phrases in text and conclude that, especially for noisy data (Switchboard data from Treebank 3), the shallow parser does much better. They also propose that a possible direction for improvements in full parsing in the near future is to combine different levels of partial parsers.

A.4.3 Conclusions

Partial parsing is a syntactic analysis approach less ambitious than full parsing, which is adequate for a variety of applications which do not need complete parse trees, but increased accuracy and speed. Psycholinguistic evidence has motivated early heuristic-based systems, which have set the stage for a variety of technologies used currently. Current emphasis within partial parsing solutions is firmly directed toward machine learning.

Systems with the highest performance are expected to use a combination of symbolic and statistical approaches, and achieve precision and recall between 90-95%. Partial parsing is a viable alternative to full parsing, and even promises to be a building block for the full parsing systems of the future.

Most acronym expansions occur within syntactic constituents in text (the example in figure 5.2(b) represents an unusual exception). Chunking of the input is a prerequisite of the acronym acquisition system presented in chapter 5, since all acronyms candidates are matched against all chunks in the text, and parameters of the match are learned. Parsed text (Penn treebank, [91]) is used in the evaluation of the system, although partial parsers have been used by the author, in unpublished work, leading to similar results.

The main role of chunking for acronym acquisition is in the reduction of possible matches per sentence (consider S tokens per sentence), from $\Omega(S^2)$ (without chunking; the number of contiguous substrings) to $O(S \log S)$ with chunking, which reduces the computational complexity of the algorithm and improves precision, since unlikely matches (which cross syntactic constituent boundaries) are not even considered for training.

A.5 Discourse and Anaphora Resolution

Discourse is a higher level of aggregation of language communication than the sentence (or utterance). The descriptive power of syntax and sentence-level semantics is not enough to explain all relationships between entities participating or referred in a discourse. The following examples are taken from [77].

- "A man whistles. A dog follows him." While syntactic information can be used to determine that the pronoun "him" cannot refer to "dog" (a reflexive pronoun is required), the meaning of the pronoun cannot be properly determined in the absence of the first sentence.

- "↗ *That frog* is the prince of Buganda. He is under a spell", where ↗ is taken to mean the 'act of pointing' [likely toward a frog]. Non-language (visual) situational information is required here for the selection of the pointed entity.

A precise semantics for natural language was first described by Montague, [101] and is being referred, since, as "traditional" semantics, a representation formalism for natural language, where elements of short utterances are encoded in a logic representation through a well-defined series of transformations. While Montagovian semantics is considered an adequate descriptive formalism for short fragments of natural language text, or utterances, there are many practical issues that make it difficult to implement in precise computational solutions.

Montague postulates that precise representations are only possible for short "fragments" of text. It has been shown that Montagovian semantics encounters significant theoretical and practical challenges when trying to encode discourse representations. The meaning of a discourse goes beyond the sum of the meaning of its parts (e.g. sentences) and incorporates complex dependencies between semantic elements in different sentences, and reliance on world knowledge and non-language situational (pragmatic) conditions in the environment. A number of semantic theories of discourse have been developed to fill this need.

A.5.1 Theories of Discourse

The pragmatic (non-language) context of discourse is difficult to access and requires very involved representations, referring to situational context and general world knowledge. Consequently, most of the work on discourse representations focuses on accounting for the relationships between semantic constituents within the discourse.

File change semantics (FCS)

File change semantics (FCS), [56] and [57] uses *Logic Forms* (Lf), which are similar with "files", where so-called "file cards" are created for each new discourse entity, when first encountered in text (through non-definite descriptions or proper names), and updates to the file card are executed whenever the entity is being subsequently referred. In a similar fashion with "library cards", file cards are built incrementally and contain at any given moment a history of all known facts about the entity on the current card, and all of its known involvements with other entities.

The meaning of a given utterance (e.g. sentence) is represented by the "file change potential", i.e. by a function from a "file" (set of file cards) to another file. In the following example, [57]:

[$_1$A girl caught a fish.$_1$] [$_2$She ate it.$_2$]

After the first sentence, file cards for the entities 'A' and 'B', introduced through the non-definite descriptions "a girl" and "a fish" are started and stored in a file F, which is in state '1', as shown in figure A.5.1.

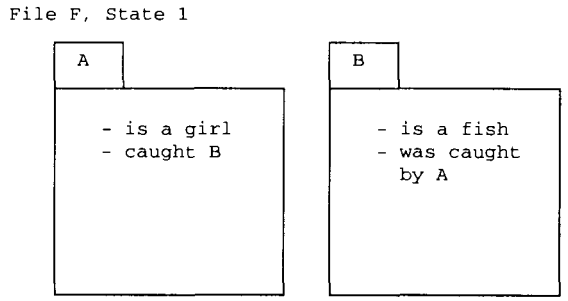


Figure A.3: File associated with the discourse: "A girl caught a fish." (Figure by author, based on the work of Heim, 1983)

After the second sentence, the file cards are updated. The file F is now in state '2', as shown in figure A.5.1.

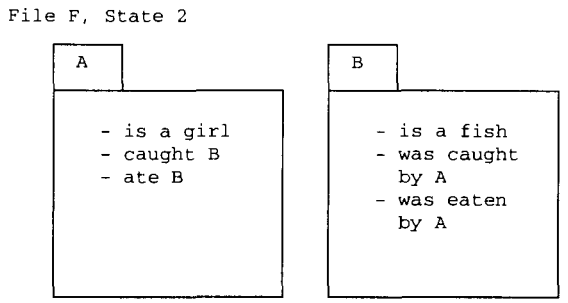


Figure A.4: File associated with the discourse: "A girl caught a fish. She ate it." (Figure by author, based on the work of Heim, 1983)

The meaning (under FCS) of the second sentence is then represented by the class of utterances that change the file F from state '1' to state '2', or by the transformation of F from '1' to '2'.

Informally, the meaning of any discourse fragment is represented by the class of utterances that can take a given file from a given initial state to a given final state, or —in the most general case—

by a function with domain the set of all possible files (initial states) and realization in the set of all possible files (final states).

Discourse structure and Centering

The theory of Discourse structure, [47] provides a framework for modeling "discourse coherence". The following components are identified: linguistic structure, intentional structure and attentional state. At the level of linguistic structure, discourse is divided into discourse segments, which exhibit internally *local coherence*, and, through relationships with other discourse segments, *global coherence*.

Within the framework of the theory of discourse structure, centering [45], [46] is a theory that relates focus of attention, choice of referring expression and perceived coherence of utterances.

Centering is supported by psycholinguistic evidence and proposes a model of coherence based on the concept of *center of attention*, or *focus* in earlier work. A measure of coherence of a discourse is given by the minimization of changes in the center of attention required from a hearer within the discourse (between different discourse segments).

Within the proposal of centering, reference ambiguity resolution, which occurs when matching is required between different candidates for a particular referent, is resolved in favor of the matching that maximizes discourse coherence or, in other words, the number of changes of center of attention required from the hearer. The following example illustrates that (sentences = discourse segments are marked with square brackets).

> $[_1$ Terry_1 really goofs sometimes. $_1]$ $[_2$ Yesterday was a beautiful day and he_1 was excited about trying on his new sailboat. $_2]$ $[_3$ He_1 wanted Tony_2 to join him on a sailing expedition. $_3]$ $[_4$ He_2 was sick and furious for being woken up so early. $_4]$

Matching 'he_2' in sentence 4 with 'Terry' is inappropriate due to the semantic mismatch between "furious for being woken up" and "excited", which leads to a match of 'he' in the last sentence with 'Tony'. Since a change of center of attention from 'Terry' to 'Tony' is requested from the hearer, the discourse lacks coherence and is peceived as *unnatural*.

Centering proposes using the distance from the current center of attention to establish the most adequate matches between referring expressions, such as pronouns, definite noun phrases, and referents, terms previously introduced through proper nouns or indefinite descriptions. All terms introduced into the discourse are entered into a sorted list of referents, a stack of so-called *forward-looking centers*, C_f.

Each referring expression has a *backward-looking center*, C_b. It will be attempted to match C_b sequentially with all the C_f-s in the stack, provided that no lexical, syntactic or semantic restrictions are violated, in an operation called *realization*. Realization results in a change of current center of attention, and possibly affects the ordering of the forward-looking centers on the stack.

Formally, [46], centering is defined based on:

- a *discourse segment*, or sequence of utterances: $DS = \{ U_i \mid i = \overline{1,n} \}$
- a partially ordered set of *forward-looking centers* for each utterance: $C_f(U_i)$
- a single *backward-looking center* for each utterance: $C_b(U_i)$

An utterance U_i *directly realizes* c (where c is a centre) if U_i is an utterance with semantic interpretation c. An utterance U_i *realizes* c if either c is an element of the situation described by U_i or c is directly realized by a subpart of U_i.

As the discourse segment is analyzed utterance by utterance, the following transitions across pairs of utterances (U_i and U_{i+1}) are defined:

- *Center Continuation*: $C_b(U_i) = C_b(U_{i+1}) = max(C_f(U_{i+1}))$, the backward center stays the same and is the highest ranked element of the current set of forward centers
- *Center Retaining*: $C_b(U_i) = C_b(U_{i+1}) \neq max(C_f(U_{i+1}))$, the backward center stays the same, but is not the highest ranked element of the current set of forward centers
- *Center Shifting*: $C_b(U_i) \neq C_b(U_{i+1})$

The following constraints on movement and realization are also presented within the framework of centering:

- $\forall c \in C_f(U_i)$, if c is realized by a pronoun in U_{i+1}, then $C_b(U_{i+1})$ must also be realized by a pronoun
- Sequences of continuation are preferred over sequences of retaining. Sequences of retaining are preferred over sequences of shifting.

A.5.2 Anaphora

The etymology of the word anaphora is Ancient Greek, a compound word with approximate meaning "the act of carrying back upstream". [49] define anaphora based on the notion of *cohesion*:

> *Anaphora* is cohesion (presupposition) which points back to some previous item.

The "pointing item" is called the *anaphor*, whereas the "pointed item" is called the *referent*, if within the current discourse or *antecedent*, when not concerned with its location.

Cataphora occurs with cohesion pointing forward to a subsequent item, cf. [97], "when a reference is made to an entity mentioned subsequently in text".

A definition which captures both anaphora and cataphora can be found in [22]:

> (...) the special case of cohesion where the meaning (sense and/or reference) of one item in a cohesive relationship (the anaphor) is, in isolation, somehow vague or incomplete, and can only properly be interpreted by considering the meanings of the other item(s) in the relationship (the antecedents).

On the other hand, [119], introduces the view that *presupposition* can be viewed as anaphora without a referent within the current discourse, which indicates a circularity of the definitions for the terms presupposition and anaphora. I suggest that *anaphora* and *presupposition* are both a manifestation of the same phenomenon of cohesion, either between terms within a given discourse (anaphora: backward oriented cohesion, cataphora: forward oriented cohesion) or between one term within and one or more terms outside a given discourse (presupposition).

It should be noted that situations where the cohesion relationship is not directional, but predominantly symmetric, are termed *coreference*.

Based on the nature of the referent, the following types of anaphora can be identified [97]. The following examples from "The Adventures of Alice in Wonderland," [21] are marked with ⌊anaphor $_{id}$⌋ and ⌈referent $_{id}$⌉:

- *Pronominal anaphora* (the most common type of anaphora) occurs when the anaphor is a personal, possessive, reflexive, demonstrative, or relative pronoun.

  > Very soon the Rabbit noticed ⌈Alice $_1$⌉, as ⌊she $_1$⌋ went hunting about, and called out to ⌊her $_1$⌋ in an angry tone (...)

- *Lexical noun phrase anaphora* occurs when when the anaphor is realized as a definite noun phrase and the referent is realized as a noun phrase.

  After these came ⌈the royal children $_2$⌉ ; there were ten of them, and ⌊the little dears $_2$⌋ came jumping merrily along hand in hand, in couples (...)

- *Noun anaphora* is the anaphoric relation between a non-lexical pro-form, such as "one" (in *one*-anaphora) and the head noun (or nominal) in a noun phrase.

  Miss, this here ought to have been a RED ⌈rose-tree $_3$⌉, and we put a white ⌊one $_3$⌋ in by mistake.

- *Verb anaphora* occurs when the referent is a verb and the anaphor an auxiliary verb, such as "did" or "have".

  The Queen turned angrily away from him, and said to the Knave '⌈Turn $_4$⌉ them over !' The Knave ⌊did $_4$⌋ so, very carefully, with one foot.

- *Adverb anaphora* occurs between an adverbial referent or Propositional Phrase (PP) and an adverb with more general meaning, such as "there" (locative), or "then" (temporal).

  Alice looked ⌊up $_5$⌋, and ⌈there $_5$⌉ stood the Queen in front of them (...)

- *Zero anaphora*, or *ellipsis* occurs when the anaphor is not present at all in the discourse, but its presence is implied. In the example below, the location of the zero-anaphor is indicated with ∅.

  At this ⌈the whole pack $_6$⌉ rose up into the air, and ⌊∅ $_6$⌋ came flying down (...)

Mitkov's book [97] is recommended for detailed definitions, comprehensive coverage of different types of anaphora, including exceptions, and many examples.

A.5.3 Anaphora resolution systems

Anaphora Resolution is one of the most difficult problems in Natural Language Processing and consists of the systematic identification of anaphor–referent relationships in text. Its difficulty comes from its necessary reliance on lower-level NLP tasks, such as syntactic analysis of discourse segments, part-of-speech disambiguation, or lexical dependence relationships. Even robust methodologies used in these prerequisite components lead to error compounding effects, diminishing the performance of anaphora resolution.

I present further a small number of categories of approaches to anaphora resolution, in a list which is by no means exhaustive. Performance figures are not presented, since, for most systems, there is no common evaluation measure. [97] is suggested for in-depth comparative analysis of the current state-of-the-art in anaphora resolution.

Centering-based systems

A number of systems, such as BFP (Brennan-Friedman-Pollard), [16] approach anaphora resolution through the framework of centering theory, with some extensions.

Tetreault [133] presents a recent implementation within centering, with increased psycholinguistic plausibility and evaluates a number of related systems (including BFP) on a common task. Teterault's work also suggests the possibility of improving theoretical models (such as centering) based on empirical evidence extracted from evaluation of "fitness at task", rather than solely through linguistic discovery methods.

Heuristic systems: full syntactic analysis

Hobb's naïve approach for pronoun anaphora, [59], [60] is perhaps one of the most influential anaphora resolution algorithm designed before 1990, and its accuracy is still comparable to many newer systems. It uses a number of heuristics, starting from completely parsed discourse segments (groups of sentences), working higher from occurrences of anaphor candidates toward sentence (S) nodes, and, from there, visiting NP descendent nodes in a breadth-first search. Nodes that violate syntactic or agreement restrictions are discarded, and the highest ranking surviving NP is assigned to be the referent.

Hobbs' algorithm was run only manually initially, and later implementations have been mainly directed at benchmarking newer anaphora resolution algorithms. It is surprising that in work as recent as [133], where a number of algorithms are compared on selections of text from the Penn Treebank corpus covering two different domains (New York Times stories and fictional texts), the performance of Hobbs' algorithm is shown to be very close to the performance of Tetreault's own LCS algorithm and better than all other algorithms evaluated.

The RAP (Resolution of Anaphora Procedure) algorithm [81] relies on McCord's slot-parser to generate syntactic representations. Referent candidates are scored based on a number of salience parameters, computed from syntactic structure and from a simple dynamic model of attentional state. RAP has remarkably high performance and signifies perhaps a shift in anaphora resolution toward

increased efficiency in real-life applications, at the expense of less ambitious linguistic models. Of concern is the fact that the input to the parser is manually "enhanced" so that inadequate parses will not be produced.

The ROSANA system, [130] is built upon a recognition of limitations of current state-of-the-art full parsing technology, and uses heuristic methods to circumvent some of them. The resulting algorithm is an implementation of the binding principles in Chomsky's "Government and Binding" framework, [25].

Heuristic systems: partial syntactic analysis

Since, especially in certain domains, full parsing of sentences is computationally difficult, prone to errors and open to an exponential explosion of the number of ambiguous parses with the length of the sentence or is simply not available, a number of approaches try to increase the robustness of anaphora resolution through the use of partial parsing methods. Within this category, [70] propose an extension to RAP that does not use a parser, but a part-of-speech analyzer, which also suggests syntactic category. The resulting system has reduced requirements in terms of processing power and resources needed for parsing, at the expense of, what the authors suggest, "only a small compromise in the quality of the results".

[69] relies on input with even less syntactic information for nominal anaphora resolution. Syntactic preference (heuristic, comparable with Hobbs' naïve algorithm) and semantic context (matching of the semantic hierarchy of the anaphor with semantic hierarchies of the referent candidates) are the only factors used within a limited text window. The performance of the system is stated to be lower, as expected, compared to Kennedy and Boguraev's system or RAP, but proves to be adequate when included in the special-purpose Information Extraction system FAUSTUS, [5].

In the same category is Mitkov's "robust, knowledge-poor algorithm" MARS, [96], which achieves remarkable performance on text from the domain of technical manuals, using a number of heuristics applied to text that is pre-processed by a part-of-speech tagger and shallow parser (identification of NPs).

Machine learning systems

In more recent developments, machine learning has been applied to anaphora resolution in a number of systems. Feature vectors are usually defined for occurrences of anaphora in tagged corpora. The feature vectors are used to induce decision-tree based classifiers, which are used subsequently to

identify anaphor-referent pairs from new text.

The RESOLVE system, [93], the system of [4] for anaphora resolution in Japanese and the system of [126] for coreference resolution of noun phrases are all built using C4.5 decision trees, and provide encouraging results.

NP coreference resolution is approached in [20] as a clustering (a form of machine learning) problem, in a "largely" unsupervised algorithm, claimed to offer promise for domain-independence, because of its independence from annotated corpora.

Using cross-language information

Aligned bilingual annotated corpora can be used as an alternative to syntactic and semantic knowledge, in order to improve upon the performance of existing knowledge-poor systems. The performance of the COCKTAIL system, [55], is improved for both English and Romanian, after training on an annotated English-Romanian corpus.

Improvement of Mitkov's MARS system is obtained, [98], described in [97], in conditions of "mutually enhancing" the performance of a single multi-lingual system on English and French versions of the same document.

Using statistical information

The statistical distribution of collocation patterns is used by [33] in generating selectional constraints in pronomial anaphora resolution. Using a similar mechanism, RAP's salience parameters are enhanced with statistical collocation distribution information. The resulting system, RAPSTAT, [32], described in [81], achieves improved accuracy, to 2%, compared to that of the RAP algorithm.

The selection phase of Hobbs' naïve algorithm is enhanced in [42] with probabilistic scores learned from an annotated corpus. These scores represent the probability for a specific anaphor to occur in a given syntactic position, the probability of antecedents proposed by Hobbs' algorithm to occur within the corpus (a so-called "mention count"), and is based on additional features such as gender, number and animacy.

A.5.4 Conclusions

Discourse analysis raises a series of new challenges compared to work at the sentence level. One of the major difficulties lies in the management of co-referring expressions. On a formal level, a

number of theories which offer accounts of such phenomena, have been advanced, such as File Change Semantics (FCS).

Psycholinguistic evidence of discourse coherence is also supportive of the theories of Discourse Structure and Centering, which offer functional models of attention and intention of participants in discourse. Anaphora is one such discourse cohesion phenomenon, and the related anaphora resolution problem is one of the most difficult problems of NLP.

A variety of algorithms for anaphora resolution have been proposed with significant performance gains still in the uncertain future. Theoretical algorithms within the centering framework, heuristics, machine learning, statistical information and cross-linguistic knowledge are used by various systems. The present is dominated by highly integrated systems achieving maximum performance, in parallel with simple, robust systems with ever-increasing baseline performance.

Long distance acronym definitions are shown in chapter 5 to be a form of discourse cohesion (hypothesis 5.4), either anaphora (when the acronym follows the expansion) or cataphora (when the acronym precedes the expansion). It follows that acronym acquisition is a special, restricted case of the anaphora resolution problem.

The rule-based acronym acquisition system presented in chapter 3 uses linguistic matching heuristics to identify expansion candidates, heuristics which are much more restrictive than –for example– pronominal anaphora matching restrictions of gender and number, and lead, consequently to much higher performance than pronominal anaphora resolution systems.

The hybrid acronym acquisition system presented in chapter 5 was also inspired by centering-based approaches to anaphora resolution, where acronym candidates not matching a given expression (through the acronym-expansion matching algorithm presented in chapter 4) are discarded (similar with gender and number-filtering) and the ordering of the forward-looking centers (matching expansion candidates for an acronym) is maintained (for acronym-expansion matches through match parameters). An important difference from centering is that machine learning is used for acronym acquisition, as for a number of reviewed anaphora resolution or noun phrase coreference systems.

A.6 Word Sense Disambiguation

The problem of word sense disambiguation (WSD) is one of the oldest problems in NLP, a prerequisite to the solution of many other NLP problems, and one of the most difficult ones, as termed in [63]:

> The problem of word sense disambiguation has been described as "AI-complete," that

> is, a problem which can be solved only by first resolving all the difficult problems in artificial intelligence (AI), such as the representation of common sense and encyclopedic knowledge.

In spite of commercial pressure, no lack of interested researchers and new systems, tangible advancement in the field is still to come.

A.6.1 Word senses

One of the main difficulties of word sense disambiguation lies in the fluid definition of word sense, which is, by no means, exact or universally agreed upon. There are two extreme opposing points of view on word sense:

1. A word sense is a stand-alone property of words, which dictates the ways in which the word can be used in context.
2. The senses of a word are represented by the possible uses of the word in context, following [39], pg. 190:

> > The use of the word 'meaning' is subject to the general rule that each word, when used in a new context is a new word.

Beyond these conflicting views, lexica are not homogenous in terms of the "granularity" of word senses. A widely used example is related to the senses of the word *bank*, [148]:

(I) *Bank* - REPOSITORY

- (I.1) Financial Bank
  - (I.1a) - an institution
  - (I.1b) - a building
- (I.2) General Supply/Reserve/Inventory

(II) *Bank* - GEOGRAPHICAL

- (II.1) Shoreline
- (II.2) Ridge/Embankment

Senses (I) and (II) above are much further apart than (I.1a) and (I.1b). [53] even mentions that (I) and (II) represent different words, with common spelling but different etymology, as opposed to different senses of the same word.

The relationship between different senses of one word is not universal, as illustrated by the difference between flat (enumeration) or very shallow sense hierarchies in most dictionaries directed at human users and hierarchical sense networks, such as WordNet.

[53] suggests that word senses (or meanings) are represented by simultaneous activations of a number of "sense potentials" for each individual occurrence of a given word in text. In other words (speaking of "word meanings"), that:

> In the everyday use of language, meanings are events, not entities.

Hanks further proposes that the appropriate question to ask is not: "What sense does this word have in this text ?", but rather: "What is the unique contribution of this word to the meaning of this text ?"

Even more complications arise when one tries to differentiate between a word sense and the use of *metaphor*, represented by, [68], pg. 623 (emphasis in original):

> (...) situations where we refer to, and reason about, concepts using words and phrases whose meanings are appropriate to *completely different kinds of concepts.*

Similarly, the *metonymy* (reference to a concept through a closely related concept), and the *synecdoche* (substitution of a part for the whole) have blurry boundaries with senses of the same word, in a relationship which is dynamic and many times predominantly historic.

Since task-specific resources are generally not available or prohibitively costly, word senses are usually determined from pre-existing NLP sources, such as:

- Dictionaries and Thesauri: general or specialized.
- Corpora: annotated or raw.
- Multilingual resources: bilingual dictionaries and bilingual or multilingual aligned corpora, such as the English-French Hansard corpus of Canadian parliamentary debates.
- Knowledge hierarchies: from (at the lowest level) domain tag/subject codes, such as the Longman Dictionary of Contemporary English (LDOCE) tag codes, through concept hierarchies, such as WordNet, to specialized knowledge bases or domain ontologies.

A.6.2 WSD evaluation

WSD is usually not an end in itself, but an essential component (and many times an important bottleneck) of systems with a specific use, such as Machine Translation, Information Extraction, Information Retrieval, etc.

The last decade has seen a lot of interest directed to WSD, with a special issue of the ACL journal, *Computational Linguistics*, [63], two combined special issues of the *Computers and the Humanities*, [71], two special issues of the *Journal of Natural Language Engineering*, vol.5, no. 2, 1999 and, upcoming, vol.9, no. 1, 2003, and also the first two instances of the SENSEVAL[7], multilingual word sense disambiguation system evaluation events.

SENSEVAL is a common evaluation framework for WSD and has consisted so far of two separate events, SENSEVAL-1 (Philadeplhia, USA, 1998) including English, French and Italian, and SENSEVAL-2 (Toulouse, France, 2001), with evaluation on twelve languages: Basque, Chinese, Czech, Danish, Dutch, English, Estonian, Italian, Japanese, Korean, Spanish, Swedish.

Participating systems are given common sense-annotated training data and are tested on common testing data. Two distinct evaluation strategies are used:

- *All words*: where all words in the testing corpus (or all open-class words) need to be tagged; this type of evaluation was only used at SENSEVAL-2.
- *Lexical sample*: where a subset of words occurring in a corpus and a subset of occurrences is used for evaluation; this type of evaluation was used at both SENSEVAL events. While better performance figures are expected using this evaluation strategy, this comes at the expense of high cost of sense tagging.

A fully-automated evaluation strategy, [41], creates artificial sense ambiguities by replacing all occurrences of two words with distinct meanings with one ambiguous marker, or *pseudoword*. Each occurrence of the pseudoword will be "sense" tagged with the original word it replaced. For example, all occurrences of the words *banana* and *door* in a text are replaced with the pseudoword *banana_or_door*. Every occurrence of the pseudoword that replaces an occurrence of "banana" in the original text will be tagged with the *sense* "banana"; similarly for "door".

An interesting differentiation of WSD, compared to other NLP problems is its intrinsic empirical nature. Even purely symbolic algorithms, in order to be meaningful, must have access to large

[7]On the web http://www.senseval.org (February 2004).

quantities of information in different types of repositories, information which is not absolute, or to the validity of which different human judges do not agree.

Even beyond that, the nature of the evaluation of WSD systems is empirical in nature, there being no absolute "gold standard" for their performance, the only possibility being comparative evaluation with imperfect human performance, either previously recorded, or collected in an experimental environment.

A.6.3 Classification of WSD systems and technologies

It is now generally agreed that the problem of WSD shows systematic differences from the problem of part-of-speech (POS) disambiguation, calling for the use of completely different algorithms. Nevertheless, some successful WSD systems either rely on preliminary POS tagging, or use components built as POS systems, such as ones using transformation-based learning.

An extreme point of view is that all WSD systems with sufficient coverage must exhibit one form or another of machine learning: learning the context of words to be disambiguated, learning the meaning of different senses of ambiguous words, or both. Consequently, a way to classify WSD systems, [148] is based on the nature of the learning: *supervised* or *unsupervised.*

Supervised systems are trained on sense-annotated corpora, such as SemCor, or HECTOR, whereas unsupervised systems are only trained on raw text. It is safe to consider that most state-of-the-art WSD systems currently rely on information from machine-readable dictionaries, with an overwhelming majority using WordNet.

Further classification is possible, but I choose to indicate that most recent high-performance WSD systems use a variety of word occurrence and definition information sources, coupled with simple semantic information.

The following technologies, and combinations thereof are present in many systems:

- *Statistical models*: [30] use HMMs for their "all-words" system submitted to SENSEVAL-2.
- *Classification*: Bayesian, entropy or expectation-maximization are widely used techniques.
- *Machine learning*: decision trees, decision lists and feature vectors are common to many systems.
- *Heuristic and transformation-based methods*: [94] use a combination of heuristic and statistical methods iteratively in the submission to SENSEVAL-2.

- *Memory-based systems*: [61] combines memory-based learning: TiMBL and rule-induction: Ripper in their submission to SENSEVAL-2.

Large integrated systems using a variety of technologies are not uncommon, such as the "John Hopkins University" system[8] submitted to SENSEVAL-2, which uses a combination of six heterogenous supervised learning subsystems, glued together using classifier combination.

A new, increased emphasis to rule-based systems, and integration of statistical systems with rule-based systems suggested [143] after the recent "swing of the pendulum" possibly too far in favor of empirical approaches to WSD.

A.6.4 Quantifying recent WSD advancement

In terms of the performance of systems presented at SENSEVAL-1 vs. SENSEVAL-2, there is no tangible sense that the state of the art has advanced significantly. A direct judgment is difficult to make, since test data is different for SENSEVAL-1: a portion of the HECTOR corpus, and SENSEVAL-2: manually tagged portions of the Penn Treebank corpus.

Nevertheless, some considerations can be made indirectly, since both evaluations use as a baseline different versions of the Lesk WSD algorithm, [83]. The highest performing such baseline, "Lesk-corpus", uses statistical information learned from words occurring in the evaluation corpus.

Since this baseline is measured for both events, I try to estimate the advancement compared to this baseline between the the best system in SENSEVAL-2 for the task "English Lexical Sample - Coarse-grained Scoring" and the best system overall in SENSEVAL-1.

[72] indicate for SENSEVAL-1 precision, p_1, and recall, r_1 of 78% for the best supervised system vs. precision, $\overline{p_1}$, and recall, $\overline{r_1}$, of 70% for the baseline estimated from graphs presented ibidem.

Data available on the SENSEVAL-2 web site[9] indicates precision, p_2, and recall, r_2, of 71.3% for the best performing system, precision, $\overline{p_2}$, and recall, $\overline{r_2}$ at 58.7% for the baseline. The preceding values are measured on the coarse-grained scoring.

I calculate the *advancement from baseline* between SENSEVAL-1 and SENSEVAL-2, $A_{1\to 2}$, as:

[8]David Yarowsky, Silviu Cucerzan, Radu Florian, Charles Schafer and Richard Wicentowski, description on the web: http://www.cogs.susx.ac.uk/lab/nlp/mccarthy/SEVALsystems.html#jhu-eng (February 2004).

[9]On the web http://www.cogs.susx.ac.uk/lab/nlp/mccarthy/SEVALsystems.html (February 2004).

$$A_{1\to 2} = \frac{p_2 - \overline{p_2}}{1 - \overline{p_2}} - \frac{p_1 - \overline{p_1}}{1 - \overline{p_1}} = 3.84\% \tag{A.4}$$

It should be noted that this value indicates relative advancement compared to the difference between baseline performance and a human golden standard, and not a 3.84% improvement in the precision and recall of systems. Reported to the values of precision and recall at SENSEVAL-2, the improvement is smaller, below 2%, in an area below 80%.

Even if the modest calculated value (3.84%) in (A.4) above presents an inevitable simplification of an otherwise complex set of differences between complex systems, measured in complex conditions, it is used here as part of a quantitative argument in support of the view that current WSD technology has reached a point where significant advancement are only possible through a significant paradigm shift.

A.6.5 Advantages and limitations of a "component" approach to WSD

[143] discusses major differences between the tasks of WSD and part-of-speech (POS) tagging, concluding that WSD does not exhibit the standard characteristics of a problem solvable by the traditional empirical paradigm in NLP:

> From these differences, of POS and WSD, I will conclude that WSD is not just one more partial task to be hacked off the body of NLP and solved.

This indicates the mutual dependencies between WSD and other NLP tasks, such as complete discourse and semantic analysis, which need to rely on word sense information, but which, in turn, must provide semantic context information to a component WSD task. Unfortunately, such complete solutions present significant software engineering challenges and are beyond our current technological capabilities.

A possible alternative is to disregard "word sense" information altogether, and focus on a clustering task of uses of words in text, such as in [123], who uses a semantic clustering method, based on a vector model, with a dimensionality reduction mechanism similar with latent semantic indexing.

It is difficult to evaluate the accuracy of such a clustering system, since overlap of "sense clusters" with documented word senses is not necessary or quantitatively predictable.

Alternate evaluation methods focus on the "fitness for the task" of larger systems such as machine translation or information retrieval, relying on a WSD component. [79] show quantitative

empirical evidence that information retrieval recall can be improved if word sense information is available for short queries.

A.6.6 Conclusions

Current WSD systems can provide good accuracy for a portion of words in text. [95] achieve 92.2% accuracy on 55% of nouns and verbs in text, by trading off recall for a boost in precision. Even higher figures are reported in earlier literature for small number of polysemous words.

Simple but powerful heuristics, such as *one sense per collocation* and *one sense per discourse*, [147] provide surprisingly good accuracy, above 96%, on small groups of polysemous words.

70-75% precision and recall on newspaper English language text can be expected for coarse word sense disambiguation for the best performing all-words systems which use both machine-readable dictionaries and are trained on tagged corpora. Higher precision and recall are reported for other domains of usage, different sets of sense tags or other languages.

Lower precision and recall, by 10-15% can be expected for the best performing systems that are not trained on tagged corpora.

Relatively high baseline figures are obtained by comparably simple, well-established, modular and extensible algorithms such as Lesk [83].

The highest performance for current systems and the promise for immediate future improvement comes from a high level of integration, using combinations of statistical methods, machine learning, heuristic and rule-based approaches.

The performance of complex systems with WSD components provides better "fitness for task" evaluation frameworks for WSD in the near future.

Acronym disambiguation is considered in chapter 7 to be a restricted case of the word sense disambiguation problem (section 7.1), where acronym forms are words and expansions are senses. The restrictive nature of acronym disambiguation arises due to the precise nature of acronym senses, which are exactly expansions. A machine learning algorithm (similar with general word sense disambiguation algorithms that learn sense usage contexts) is presented, which results in figures that are unexpectedly high for general word sense disambiguation.

Methods for automatically acquiring training examples from the results of web search engine queries and for automatic evaluation are also presented, which provide inexpensive solutions to the highest cost items in word sense disambiguation systems.

A.7 Field Matching and String Alignment

Field matching is a problem occurring in databases where different record fields (usually text) are variants of the same form, which refers to a unique entity. An alternative formulation, [11]:

> In general the field matching problem is to determine whether or not two field values are syntactic alternatives that designate the same semantic entity.

The field matching problem (ibid.) is also refered as: record linkage, the merge/purge problem, duplicate detection, hardening soft databases, and reference matching.

For text fields, variants occur due to alternate acceptable spellings, alternate acceptable word order, misspellings, abbreviations, or even semantic equivalence.

A.7.1 Statistical foundations

The field matching problem originates in statistics [108], where automatic relationships are drawn between multiple records, in order to identify multiple entries for the same individual or family.

A formal treatment of field matching is proposed by Fellegi-Sunter [38], where three classes of matching fields are introduced: link, non-link and possible link, and probabilistic vector functions are associated with the different types of errors. The problem of matching is reduced to the minimization of a cumulative error function associated with performance on a complete data set.

Fellegi-Sunter present a mathematical apparatus (theorem and corrolaries) which allow for computationally feasible solutions to a number of practical instances of the field matching problem. The main limitation, which has still not been overcome in the most general case, is the size of the possible matching space, quadratic in the number of records; for example, for 10^{11} records, the cummulative size of the matching search space is 10^{22}, beyond the limit of current computing power and data storage capacity. The theoretical reason behind this limitation lies in the fact that metrics of the matching search space as described in section A.7.3, generally impose only *partial ordering*, which require exhaustive search (in quadratic time) for an optimum solution.

More recently, machine learning techniques, such as expectation-maximization are used in conjunction with the Fellegi-Sunter model for applications such as census, [144]. Recent needs, such as electronic security and the management of privacy and disclosure of information, represent new flavors of the problem.

A.7.2 Longest common substring

String matching is a central requirement for solutions to the field matching problem. In the simplest case, field matching consists of repeated string matching (also called *string alignment*) operations applied to pairs of fields from the matching space.

String alignment, in one of its simplest formulations is known as the *longest common substring* or *longest common subsequence* (LCS) problem, where the a substring (with possible discontinuities) of maximum length is to be found for two given strings A and B of lengths n and m.

It is immediately obvious that enumerating all discontiguous substrings of A and B and comparing them pair by pair (the brute force approach) is not feasible in general: complexity $\Omega(2^{nm})$.

A "folk" dynamic programming algorithm [28] solves the LCS problem by building a two-dimensional grid with the two strings as the horizontal, respectively vertical dimensions, as shown in figure A.5. The objective is to navigate from the North-West to the South-East corners of the grid, by following only borders of cells (South or East), or, if the letters of the two strings in the same column, respectively row as the cell match, diagonal movement South-East is also possible. The algorithm builds the path (bold in the figure) which maximizes, for any given point on the grid, the number of matches (diagonal moves) executed up to that point.

The execution of the algorithm also results in a so-called *alignment* of the two strings, as shown in (A.5), where '-' (dashes) indicate shifts of the current character in each string, and '|' indicate matching pairs of characters.

```
SP-ORT-
|  ||
S-COR-E
```
(A.5)

An equivalent formulation builds an array $M[0..n][0..m]$, where each cell $M[i,j]$ represents the *matching score* (the length of the longest common substring) of the substrings $A[1..i]$ and $B[1..j]$. After initializing the North row and West column to zero, elements in the array are consecutively calculated as:

$$M[i][j] = \begin{cases} max(M[i][j-1], M[i-1][j]) & \text{if } A[i] \neq B[j] \\ M[i-1][j-1]+1 & \text{otherwise} \end{cases} \quad \text{(A.6)}$$

The value calculated by the algorithm for cell $M[n][m]$ (South-Eastern-most cell) is the length

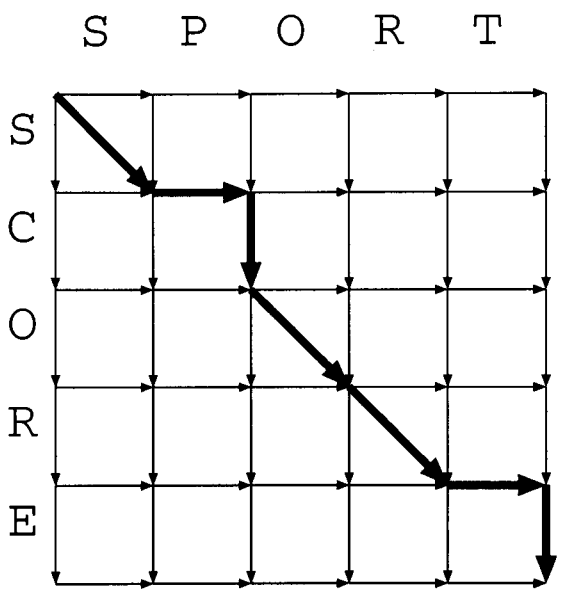


Figure A.5: Running of the *longest common substring* algorithm for the strings "SPORT" and "SCORE" (Figure by author)

of the longest common subsequence, which can be recovered by backtracking on the successive choices made by equation (A.6), starting from the South-East corner. The algorithm works because of the *optimum substructure* of the length of the longest common subsequence, [28, Chapter 15].

A.7.3 Distance metrics in string space

The longest common subsequence algorithm can be perceived as an optimization problem, where the maximum value of a *string distance* metric (number of matching characters) for two strings is calculated. Other string distance metrics have also been used, as follows.

A heuristic string comparison metric, designed to accomodate spelling variations: *soundex*, is used in early field matching solutions [108] and still offered as a function in commercial database products.

String-edit distance, also called *Levenshtein* distance [84] is the minimum number of insertions, deletions or substitutions that can transform a source string into a target string. The "longest common subsequence" metric can be perceived as a restricted case of string-edit distance, where substitutions are not allowed (and consequently counted). The calculation of the string-edit distance is achieved by a dynamic programming algorithm also very similar with the LCS algorithm.

A generalization of Levenshtein distance, called *Needleman-Wunsch* distance, [106] associates weigths with the basic string operations (insertion, deletion and substitution). The resulting dynamic programming algorithm is very similar with string-edit distance calculation and LCS.

Needleman-Wunsch distance and its variants have been widely used in molecular biology problems associated with genomic or proteomic strings. Biologically motivated *affine penalties* that associate different weigths for the deletion of longer subsequences are also used by Smith and Waterman, [125]. In this case, values in a third matrix are calculated by the algorithm. Numerous variants of the sequence alignment are used in molecular biology, in problems such as geometrical structure prediction, phylogeny, multiple and inexact matching, etc. Gusfield [48] provides a comprehensive coverage of string algorithms, from a bioinformatics perspective. An earlier brief summary [104] also focuses on algorithms with molecular biology applications. An earlier overview of string alignment techniques, [120] also discusses applications in areas such as speech recognition.

String alignment problems have in common an *optimum substructure* property, which leads to implementations dramatically more efficient (polynomial time) than brute force complete searches of the matching space (exponential time in most cases).

Other string distance metrics can be applied to multi-word expressions, even full documents.

Cosine similarity, [118], widely used in information retrieval applications, is such a metric, based on normalized distance between vectors in multi-dimensional feature spaces.

A.7.4 Field matching algorithms

Statistical information has been used starting with field alignment problems starting with their earlier instantiations, [108], [38].

Building upon this foundation, more recent systems have used various machine learning methodologies such as expectation maximization, [144]. Expectation Maximization (EM) algorithms are also used, [116] to learn transition probabilities in a stochastic transducer that calculates probabilistic Levenshtein-like string distances. Similarly transition probabilities are learned (using EM and also support vector machines), [11] for modeling distance metrics for Smith and Waterman-like string alignment with gaps.

Various problem-specific heuristics are also used, [99], as well as clustering (union/find), [100] for the identification of alternate spellings, primarily abbreviations.

A.8 Conclusions

Field matching is an old, well studied problem in computer science, which relies on the calculation of string distance metrics, using a variety of approaches, with popularity in diverse domains such as molecular biology, speech recognition and information extraction.

Recent efforts use statistically-motivated machine learning methodologies, which derive parameters of matching between strings. Heuristic approaches have also been used. Applications to abbreviation normalization (which can be reformulated as a field matching problem) have been attempted.

Field matching is directly related to the acronym acquisition algorithm presented in chapter 5, where the identification of acronym definitions (acronym–expansion pairs) can be reformulated as an instance of the field matching problem.

Acronym expansion matching, as defined in chapters 4 and 6 can be viewed as an instance of the string alignment problem. The results of the acronym–expansion matching algorithm, as well as the metrics of *matching ambiguity* and *accidental match probability* (defined in chapter 6) can be viewed as string distance metrics.

I suggest that multilayered, linguistically-motivated string matching, as proposed in chapter 4 for acronym-expansion pairs, holds promise for the string distance formalisms that go beyond

the acronym or abbreviation domains, and is applicable to named entity recognition, coreference resolution, or even anaphora resolution. Following this intuition is left as future work.

Bibliography

[1] Steven Abney. Parsing by chunks. In Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, *Principle-Based Parsing*, pages 257–278. Kluwer Academic Publishers, 1991.

[2] Acronym finder. http://www.AcronymFinder.com, June 2003.

[3] E. Antworth. Morphological parsing with a unification-based word grammar. In *Proceedings of the North-Texas Natural Language Processing Workshop. University of Texas, Arlington*, pages 24–32, 1994.

[4] Chinatsu Aone and Scott William Bennett. Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL'95)*, pages 122–129, Cambridge, MA, 1995.

[5] Douglas Appelt, Jerry Hobbs, John Bear, David Israel, Megumi Kameyama, Andy Kehler, David Martin, Karen Myers, and Mabry Tyson. Sri international fastus system: Muc-6 test results and analysis. In *Proceedings of the 6th Message Understanding Conference*, pages 237–248, Columbia, MD, 1995. DARPA.

[6] Diana Arcangeli and Terence D. Langendeon, editors. *Optimality Theory: An Overview*. Blackwell Publishers, 1997.

[7] J.L. Austin. Performative utterances. In A.P. Martinich, editor, *The philosophy of language*, pages 115–124. Oxford University Press, 1985.

[8] G. Edward Barton Jr. The complexity of two-level morphology. In Jr. Barton, G. Edward, Robert C. Berwick, and Eric Sven Ristad, editors, *Computational Complexity and Natural Language*, chapter 5. MIT Press, Cambridge, MA, 1987.

[9] A.F.L. Beeston. *The Arabic Language Today*. Hutchinson & Co, 1970.

[10] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[11] Mikhail Bilenko and Raymond J. Mooney. Learning to combine trained distance metrics for duplicate detection in databases. Technical Report AI 02-296, Artificial Intelligence Lab, University of Texas at Austin, February 2002.

[12] Reinhard Blutner. Some aspects of optimality in natural language interpretation. *Journal of Semantics*, 17(3), 2000.

[13] Mary Rose Bonk and Pamela Dear, editors. *Acronyms, initialisms, & abbreviations dictionary*. Gale Group, Detroit, 2001.

[14] Antal van den Bosch. *Learning to pronounce written words. A study in inductive language learning*. PhD thesis, Universiteit Maastricht, The Netherlands, Cadier en Keer, 1997.

[15] S. Bradner. Rfc 2026: The internet standards process–revision 3. The Internet Engineering Task Force; http://www.ietf.org, February 2004, 1997.

[16] Susan Brennan, Marilyn Walker Friedman, and Carl Pollard. A centering approach to pronouns. In *Proceedings, 25th Annual Meeting of the Association for Computational Linguistics*, pages 155–162, Stanford, 1987.

[17] Joan Bresnan. Optimal syntax. In Joost Dekkers, Frank van der Leeuw, and Jeroen van de Weijer, editors, *Optimality Theory: Phonology, Syntax and Acquisition*, pages 334–385. Oxford University Press, Oxford, 2000.

[18] Gavin Burnage. Celex: A guide for users. Technical report, Centre for Lexical Information, Nijmegen, 1990.

[19] Garland Cannon. Abbreviations and acronyms in english word formation. *American Speech*, 64(2):99–127, 1989.

[20] Claire Cardie and Kiri Wagstaff. Noun phrase coreference as clustering. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 82–89, College Park, MD, 1999.

[21] Lewis Carroll. *Alice's adventures in wonderland.* Heinemann, London, 1972.

[22] David Carter. *Interpreting anaphors in natural language texts.* Halsted Press, 1987.

[23] Hamako Ito Chaplin and Samuel E. Martin. *A Manual of Japanese Writing*. Yale University Press, 1969.

[24] Ping Chen. *Modern Chinese: its history and sociolinguistics*. Cambridge University Press, 1999.

[25] Noam Chomsky. *Lectures on government and binding*. Foris Publications, Dordrecht, Holland, 1981.

[26] Noam Chomsky. *The minimalist program.* MIT Press, Cambridge, MA, 1995.

[27] Noam Chomsky and Howard Lasnik. Principles and parameters theory. In J. Jacobs, A. von Stechow, W. Sternefelt, and T. Vennemann, editors, *Syntax: An international handbook of contemporary research.* de Gruyter, Berlin, 1993.

[28] Thomas H. Cormen, Charles E. Leisserson, Ronald E. Rivest, and Clifford Stein, editors. *Introduction to algorithms.* MIT Press and McGraw Hill, 2001.

[29] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[30] Eric Crestan, M. El-Béze, and C. De Loupy. Improving wsd with multi-level view of context monitored by similarity measure. Senseval-2, Toulouse, 2001.

[31] Mathias Creutz and Krista Lagus. Unsupervised discovery of morphemes. In Michael Maxwell, editor, *Proceedings of the Sixth Meeting of the ACL Special Interest Group in Computational Phonology*, pages 21–30, Philadelphia, Pennsylvania, 2002. ACL.

[32] Ido Dagan. *Multilingual statistical approaches for natural language disambiguation (in Hebrew).* PhD thesis, Israel Institute of Technology, Haifa, Israel, 1992.

[33] Ido Dagan and Alon Itai. Automatic processing of large corpora for the resolution of anaphora references. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING'90)*, volume 3, pages 1–3, Helsinki, Finland, 1990.

[34] Hervé Déjean. Learning syntactic structures with xml. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 133–135. Lisbon, Portugal, 2000.

[35] Jason Eisner. Easy and hard constraint ranking in optimality theory. In Jason Eisner, Lauri Karttunen, and Alain Theriault, editors, *Finite-State Phonology: Proceedings of the 5th Workshop of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 22–33, Luxembourg, 2000.

[36] Interinstitutional style guide. annex 4 main acronyms and abbreviations. European Union: Publications Office; http://eur-op.eu.int/code/en/en-5000400.htm, February 2004, 2004.

[37] Christiane Fellbaum, editor. *WordNet, An Electronic Lexical Database.* MIT Press, 1998.

[38] Ivan P. Felligi and Alan B. Sunter. A theory for record linkage. *Journal of the American Statistical Society*, 64:1183–1210, 1969.

[39] John Rupert Firth. *Papers in linguistics, 1934-1951.* Oxford University Press, 1957.

[40] SFU Library Assistant for Theses. Journal article style for theses. http://www.lib.sfu.ca/researchhelp/writing/thesesformatting/articlestyle.pdf, April 2004.

[41] W. Gale, K. Church, and D. Yarowsky. On evaluation of word-sense disambiguation systems. In *Proceedings, 30th Annual Meeting of the Association for Computational Linguistics*, pages 249–256, 1992.

[42] Niyu Ge, John Hale, and Eugene Charniak. A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora*, pages 161–171, Montréal, Canada, 1998.

[43] James Paul Gee and François Grosjean. Performance structures: a psycholinguistic and linguistic appraisal. *Cognitive Psychology*, 15:411–458, 1983.

[44] John Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2), 2001.

[45] Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. Providing a unified account of definite noun phrases in discourse. In *Proc. 21st Annual Meeting of the ACL*, pages 44–50, Cambridge, Massachusetts, 1983. Association of Computational Linguistics.

[46] Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. Centering: A framework for modelling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225, 1995.

[47] Barbara J. Grosz and C. Sidner. Attentions, intentions and the structure of discourse. *Computational Linguistics*, 12:175–204, 1986.

[48] Dan Gusfield. *Algorithms on strings, trees, and sequences : computer science and computational biology*. Cambridge University Press, 1997.

[49] Michael A.K. Halliday and Ruqaiya Hasan. *Cohesion in English*. Number 9 in English Language Series. Longman, 1976.

[50] Michael Hammond. Syllable parsing in english and french. Rutgers Optimality Archive, http://roa.rutgers.edu (February 2004), 1995.

[51] Michael Hammond. Optimality theory and prosody. In Diana Archangeli and D. T. Langendoen, editors, *Optimality Theory - An Overview*. Blackwell, 1997.

[52] Michael Hammond. Parsing syllables: modeling ot computationally. Rutgers Optimality Archive, http://roa.rutgers.edu (February 2004), 1997.

[53] Patrick Hanks. Do word meanings exist? *Computers and the Humanities*, 34(1-2):205–215, 2000.

[54] Sami A. Hanna and Nagub Greis. *Writing Arabic*. The University of Utah Printing Service, 1965.

[55] Sanda Harabagiu and Steven Maiorano. Multilingual coreference resolution. In *Proceedings of Conference on Applied Natural Language Processing, North American Chapter of the Association for Computational Linguistics (ANLP-NAACL2000)*, pages 142–149, Seattle, WA, 2000.

[56] I. Heim. *The Semantics of Definite and Indefinite Noun Phrases*. PhD thesis, University of Massachussets, Amherst, 1982.

[57] I. Heim. File change semantics and the familiarity theory of definiteness. In R. Bäuerle, C. Schwarze, and A. von Stechow, editors, *Meaning, Use and Interpretation of Language*, pages 164–189. De Gruyter, Berlin, 1983.

[58] Petra Hendriks and Helen de Hoop. Optimality theoretic semantics. *Linguistics and Philosophy*, 24(1):1–32, 2001.

[59] J.R. Hobbs. Pronoun resolution. Technical Report 76-1, Department of Computer Science, City College, City University of New York, 1976.

[60] J.R. Hobbs. Resolving pronoun references. *Lingua*, 44:311–338, 1978.

[61] Véronique Hoste, Walter Daelemans, Hendrickx Iris, and Antal van den Bosch. Evaluating the results of a memory-based word-expert approach to unrestricted word sense disambiguation. In *Proceedings of the Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 95–101, Philadelphia, PA, USA, 2002.

[62] David A. Hull and Gregory Grefenstette. A detailed analysis of english stemming algorithms. Technical Report TR MLTT-023. RXRC, Rank XEROX, 1996.

[63] Nancy Ide and Jean Véronis. Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40, 1998.

[64] Requests for comments. The Internet Engineering Task Force; http://www.ietf.org, February 2004, 2000.

[65] Thorsten Joachims. Text categorization with support vector machines. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 137–142, Chemnitz, Germany, 1998.

[66] Thorsten Joachims. Making large-scale svm learning practical. In B. Schölkopf, Christopher J.C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.

[67] Christer Johansson. A context sensitive maximum likelihood approach to chunking. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 136–138. Lisbon, Portugal, 2000.

[68] Daniel Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition.* Prentice Hall series in artificial intelligence. Prentice Hall, 2000.

[69] Megumi Kameyama. Recognizing referential links: An information extraction perspective. In Ruslan Mitkov and Branimir Boguraev, editors, *Proceedings of ACL/EACL Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, pages 46–53, Madrid, 1997.

[70] Christopher Kennedy and Branimir Boguraev. Anaphora for everyone: Pronominal anaphora resolution without a parser. In *Proceedings of the 16th International Conference on Computational Linguistics, (COLING'96)*, pages 113–118, Copenhagen, Denmark, 1996.

[71] A. Kilgarriff and M. Palmer. Introduction to the special issue on senseval. *Computers and the Humanities*, 34(1-2):1–13, 2000.

[72] A. Kilgarriff and J. Rosenzweig. Framework and results for english senseval. *Computers and the Humanities*, 34:15–48, 2000.

[73] Heinz Koblischke. *Lexikon der Abkürzungen : über 50000 Abkürzungen, Kurzwörter, Zeichen und Symbole*. Bertelsmann-Lexikon-Verlag, 1994.

[74] Rob Koeling. Chunking with maximum entropy models. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 139–141. Lisbon, Portugal, 2000.

[75] Kimmo Koskenniemi. Representations and finite-state components in natural language. In Emmanuel Roche and Yves Schabes, editors, *Finite-State Language Processing*. MIT Press, 1997.

[76] Kimmo Koskenniemi and Kenneth Ward Church. Complexity, two-level morphology and finnish. In *Proceedings of the 12th International Conference on Computational Linguistics, COLING-88, Budapest*, pages 335–339, 1988.

[77] Emiel Jacques Krahmer. *Presupposition and anaphora*. Number 89 in CSLI lecture notes. Center for the Study of Language and Information, Stanford University, 1998.

[78] Robert Krovetz. Viewing Morphology as an Inference Process,. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–203, Pittsburg, Pennsylvania, 1993.

[79] Robert Krovetz and W. Bruce Croft. Lexical ambiguity and information retrieval. *Information Systems*, 10(2):115–141, 1992.

[80] Taku Kudoh and Yuji Matsumoto. Use of support vector learning for chunk identification. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 142–144. Lisbon, Portugal, 2000.

[81] Shalom Lappin and Herbert Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–562, 1994.

[82] Leah S. Larkey, Paul Oglivie, M. Andrew Price, and Brenden Tamilio. Acrophile: An automated acronym extractor and server. In *Digital Libraries '00 - The Fifth ACM Conference on Digital Libraries (San Antonio, June 2-7, 2000)*, pages 205–214. ACM Press, 2000.

[83] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine code from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, Toronto, Ontario, Canada, 1986. ACM Press.

[84] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.

[85] Xin Li and Dan Roth. Exploring evidence for shallow parsing. In Walter Daelemans and Rémi Zajac, editors, *Proceedings of CoNLL-2001*, pages 38–44. Toulouse, France, 2001.

[86] Simon Fraser University Graduate Studies & University Libraries. Regulations and guidelines for the preparation of theses, extended essays, and projects. http://www.lib.sfu.ca/researchhelp/writing/reg-gyds.htm, April 2004.

[87] Hongfang Liu, Stephen B. Johnson, and Carol Friedman. Automatic resolution of ambiguous terms based on machine learning and conceptual relations in the umls. *J. Am. Med. Inform. Assoc.*, 9(6):621–636, 2002.

[88] Hongfang Liu, Yves A. Lussier, and Carol Friedman. A study of the umls abbreviations. In *Proceedings of the AMIA 2001 Symposium*, pages 393–397. Hanley & Belfus, 2001.

[89] Julie Beth Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11:22–31, 1968.

[90] Rune B. Lyngsø, M. Zucker, and Christian N.S. Pedersen. Internal loops in rna secondary structures. In *RECOMB99: Proceedings of the Third Annual International Conference on Computational Molecular Biology*, pages 260–267, Lyon, France, 1999.

[91] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[92] John J. McCarthy and Alan S. Prince. Prosodic morphology. In J. Goldsmith, editor, *The handbook of Phonological Theory*, chapter 9, pages 318–366. Basil Blackwell, Oxford, 1995.

[93] Joseph F. McCarthy and Wendy G. Lehnert. Using decision trees for coreference resolution. In *Proceedings of the the 1995 International Joint Conference on AI (IJCAI)*, pages 1050–1055, Montreal, Quebec, 1995.

[94] Rada Mihalcea and Dan Moldovan. An iterative approach to word sense disambiguation. In *Proceedings of Flairs-2000*, pages 219–223, Orlando, Florida, 2000.

[95] Rada Mihalcea and Dan Moldovan. A highly accurate bootstrapping algorithm for word sense disambiguation. *International Journal on Artificial Intelligence Tools*, 10(1-2):5–21, 2001.

[96] Ruslan Mitkov. Robust pronoun resolution with limited knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING'98/ACL'98)*, pages 869–975, Montreal, Canada, 1998.

[97] Ruslan Mitkov. *Anaphora Resolution*. Pearson Education Limited, 2002.

[98] Ruslan Mitkov and Catalina Barbu. Improving pronoun resolution in two languages by means of bilingual corpora. In *Proceedings of the Discourse, Anaphora and Reference Resolution Conference (DAARC 2000)*, pages 133–137, Lancaster, UK, 2000.

[99] Alvaro E. Monge and Charles P. Elkan. The field-matching problem: algorithm and applications. In *The Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 267–270, August 1996.

[100] Alvaro E. Monge and Charles P. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *The proceedings of the SIGMOD 1997 workshop on data mining and knowledge discovery*, pages 23–29, May 1997.

[101] Richard Montague. The proper treatment of quantification in english. In Richmond H. Thomason, editor, *Formal philosophy : selected papers of Richard Montague*. Yale University Press, New Haven, 1974.

[102] Karin Müller. Probabilistic context-free grammars for phonology. In Michael Maxwell, editor, *Proceedings of the Sixth Meeting of the ACL Special Interest Group in Computational Phonology*, pages 70–80, Philadelphia, 2002. ACL.

[103] Krista Lynn Muller. Treating 'kre-8-ive' spellings for natural language processing, 1999. Master Thesis, Simon Fraser University, Department of Linguistics.

[104] Eugene W. Myers. An overview of sequence comparison algorithms in molecular biology. Technical Report TR-91-29, Department of Computer Science, University of Arizona, Tucson, 1991.

[105] National Libraries of Medicine. *UMLS Knowledge Sources Manual*, August 2002.

[106] Saul B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453, 1970.

[107] Goran Nenadić, Irena Spasić, and Sophia Ananiadou. Automatic acronym acquisition and management within domain-specific texts. In *Proceedings of LREC-3*, pages 2155–2162, Las Palmas, Spain, 2002.

[108] H.B. Newcombe, Kennedy J.M., Axford S.J., and A.P. James. Automatic linkage of vital records. *Science*, 130(3381):954–959, October 1959.

[109] National Library of Medicine. Medline. Available at: http://www.ncbi.nlm.nih.gov/PubMed/, February 2004, 2002.

[110] Opaui guide to lists of acronyms, abbreviations, and initialisms. http://spin.com.mx/~smarin/acro.html, February 2004.

[111] Miles Osborne. Shallow parsing as part-of-speech tagging. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 145–147. Lisbon, Portugal, 2000.

[112] Serguei Pakhomov. Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL) Philadelphia, July 2002*, pages 160–167, 2002.

[113] Ferran Pla, Antonio Molina, and Natividad Prieto. Improving chunking by means of lexical-contextual information in statistical language models. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 148–150. Lisbon, Portugal, 2000.

[114] Martin F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, July 1980.

[115] Alan S. Prince and Paul Smolensky. Optimality theory: Constraint interaction in generative grammar. Technical Report 2, Piscataway, NJ: Rutgers Center for Cognitive Science, Rutgers University, and Boulder, CO: Department of Computer Science, University of Colorado, 1993.

[116] Eric Sven Ristad and Peter N. Yianilos. Learning string edit distance. Technical Report CS-TR-532-96, Department of Computer Science, Princeton University, 1997.

[117] Haiim B. Rosén. *Contemporary Hebrew*. Mouton & Co., The Hague, 1977.

[118] Gerard Salton. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, 1989.

[119] R. van der Sandt. Presupposition projection as anaphora resolution. *Journal of Semantics*, 9:223–267, 1992.

[120] David Sankoff and Joseph B. Kruskal, editors. *Time warps, string edits, and macromolecules : the theory and practice of sequence comparison*. Addison-Wesley Pub. Co., Advanced Book Program, 1983.

[121] Edgar Scheitz. *Dictionary of Russian Abbreviations*. Elsevier, 1986.

[122] Patrick Schone and Daniel Jurafsky. Knowledge-free induction of morphology using latent semantic analysis. In Claire Cardie, Walter Daelemans, Claire Nédellec, and Erik Tjong Kim Sang, editors, *Proceedings of the Fourth Conference on Computational Natural Language*

Learning and of the Second Learning Language in Logic Workshop, Lisbon, 2000, pages 67–72, Somerset, New Jersey, 2000. Association for Computational Linguistics.

[123] Hinrich Schütze. Dimensions of meaning. In *Proceedings of Supercomputing '92, Minneapolis.*, pages 787–796, 1992.

[124] John R. Searle. Proper names. In A.P. Martinich, editor, *The philosophy of language*, pages 115–124. Oxford University Press, 1985.

[125] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[126] Wee Meng Soon, Daniel Chung Yong Lim, and Hwee Tou Ng. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):512–544, 2001.

[127] Paul Sprachman. *Language and culture in Persian.* Mazda Publishers, Costa Mesa, California, 2002.

[128] Richard Sproat. Review of "pc-kimmo: a two-level processor for morphological analysis" by evan l. antworth. *Computational Linguistics*, 17(2):229–231, 1991.

[129] Richard Sproat, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorfk, and Christopher Richards. Normalization of non-standard words. *Computer Speech and Language*, 15:287–333, 2001.

[130] Roland Stuckardt. Design and enhanced evaluation of a robust anaphor resolution algorithm. *Computational Linguistics*, 27(4):479–506, 2001.

[131] J. Sykes, editor. *The Concise Oxford dictionary of current English.* Oxford University Press, 6 edition, 1976.

[132] Kazem Taghva and Jeff Gilbreth. Recognizing acronyms and their definitions. Technical Report 95-03, ISRI (Information Science Research Institute) UNLV, 1995.

[133] Joel R. Tetreault. A corpus-based evaluation of centering and pronoun resolution. *Computational Linguistics*, 27(4):507–520, 2001.

[134] Erik F. Tjong Kim Sang. Text chunking by system combination. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 151–153. Lisbon, Portugal, 2000.

[135] Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the conll-2000 shared task: Chunking. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 127–132. Lisbon, Portugal, 2000.

[136] Erik F. Tjong Kim Sang and Hervé Déjean. Introduction to the conll-2001 shared task: Clause identification. In Walter Daelemans and Rémi Zajac, editors, *Proceedings of CoNLL-2001*, pages 53–57. Toulouse, France, 2001.

[137] Janine Toole. **A hybrid approach** to the identification and expansion of abbreviations. In *RIAO 2000 6th Conference on Content-Based Multimedia Information Access*, pages 725–736, College de **France, Paris,** France, 2000.

[138] Hans van Halteren. **Chunking** with wpdv models. In Claire Cardie, Walter Daelemans, Claire Nedellec, and **Erik Tjong Kim Sang**, editors, *Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal*, **pages 154–156,** 2000.

[139] Vladimir Vapnik. ***Statistical Learning Theory***. Wiley, New York, Chichester, 1998.

[140] Vladimir Vapnik. ***The Nature of Statistical Learning Theory***. Springer, 2000.

[141] Jorn Veenstra **and Antal van den** Bosch. Single-classifier memory-based phrase chunking. In Claire Cardie, **Walter Daelemans,** Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal*, pages 157–159, 2000.

[142] Marc Vilain and **David Day. Phrase** parsing with rule sequence processors: an application to the shared conll **task. In Claire Cardie,** Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal*, pages 160–162, 2000.

[143] Yorick Wilks. Is word **sense** disambiguation just one more nlp task? *Computers and the Humanities*, 34:**235–243, 2000.**

[144] William E. **Winkler and Yves Thibaudeau.** An application of the fellegi-sunter model of record linkage to **the 1990 u.s. census.** Technical Report RR91/09, U.S. Bureau of the Census, 1991.

[145] World wide web acronym **and** abbreviation server (wwwaas). http://www.ucc.ie/cgi-bin/acronym, June 2001.

[146] Jinxi Xu and Bruce W. **Croft. Corpus-based** stemming using co-occurrence of word variants. *ACM Transactions on Information Systems*, **16**(1):61–81, 1998.

[147] David Yarowsky. Unsupervised **word sense** disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Boston, **Massachussets, 1995.**

[148] David Yarowsky. Word-sense **disambiguation.** In Hermann Dale, Robert Moisl and Harold Somers, editors, *Handbook of natural language processing*, pages 629–654. Marcel Dekker, New York, 2000.

[149] Stuart Yeates. Automatic **extraction of acronyms** from text. In *Proceedings of the Third New Zealand Computer Science Research Students Conference Hamilton, New Zealand, April 1999, University of Waikato*, **pages 117–124, 1999.**

[150] Stuart Yeates, David Bainbridge, **and Ian H. Witten.** Using compression to identify acronyms in text. Submitted to Data **Compression** Conference DCC 2000, Snowbird, Utah, 2000.

[151] M. Yoshida, K. Fukuda, and T Takagi. Pnad-css: a workbench for construction a protein name abbreviation dictionary. *Bioinformatics*, 16:169–175, 2000.

[152] Manuel Zahariev. Automatic acquisition of long-distance acronym definitions. In *Proceedings of HIS'03 Hybrid Intelligent Systems, Melbourne 2003*, pages 582–592. IOS Press, 2003.

[153] Manuel Zahariev. An efficient methodology for acronym-expansion matching. In Nazli Goharian, editor, *Proceedings of the International Conference on Information and Knowledge Engineering, IKE'03*, volume 1, pages 32–37. CSREA Press, 2003.

[154] Manuel Zahariev. A linguistic approach to extracting acronym expansions from text. *KAIS: Knowledge and Information Systems*, 6(3):366–373, 2004.

[155] GuoDong Zhou, Jian Su, and TongGuan Tey. Hybrid text chunking. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 163–166. Lisbon, Portugal, 2000.

[156] Pierre Zweigenbaum and Natalia Grabar. Automatic acquisition of morphological knowledge for medical language processing. *Artificial Intelligence in Medicine, Lecture Notes in Artificial Intelligence*, pages 416–420, 1999.

[157] Pierre Zweigenbaum and Natalia Grabar. Automatic acquisition of domain-specific morphological resources from thesauri. In *Actes de RIAO 2000 : Accès à l'Information Multimédia par le Contenu*, pages 765–784, Paris, France, C.I.D., 2000.