Rune Sætre

GeneTUC: Natural Language Understanding in Medical Text

Doctoral thesis
for the degree of doktor ingeniør

Trondheim, June 2006

Department of Computer and Information Science
Norwegian University of Science and Technology (NTNU)
NO-7491 Trondheim, Norway

NTNU
Norwegian University of Science and Technology
Doctoral thesis
for the degree doktor ingeniør
Faculty of Information Technology,
Mathematics and Electrical Engineering
Department of Computer and Information Science

ISBN 82-471-7866-4 (printed version)
ISBN 82-471-7865-6 (electronic version)
ISSN 1503-8181

Doctoral thesis at NTNU, 2006-59

This is version 1.0, May 29, 2006.

Abstract

Natural Language Understanding (NLU) is a 50 years old research field, but its application to molecular biology literature (BioNLU) is a less than 10 years old field. After the complete human genome sequence was published by Human Genome Project and Celera in 2001, there has been an explosion of research, shifting the NLU focus from domains like news articles to the domain of molecular biology and medical literature. BioNLU is needed, since there are almost 2000 new articles published and indexed every day, and the biologists need to know about existing knowledge regarding their own research. So far, BioNLU results are not as good as in other NLU domains, so more research is needed to solve the challenges of creating useful NLU applications for the biologists.

The work in this PhD thesis is a "proof of concept". It is the first to show that an existing Question Answering (QA) system can be successfully applied in the hard BioNLU domain, after the essential challenge of *unknown entities* is solved. The core contribution is a system that discovers and classifies unknown entities and relations between them automatically. The World Wide Web (through Google) is used as the main resource, and the performance is almost as good as other named entity extraction systems, but the advantage of this approach is that it is much simpler and requires less manual labor than any of the other comparable systems.

The first paper in this collection gives an overview of the field of NLU and shows how the Information Extraction (IE) problem can be formulated with *Local Grammars*. The second paper uses *Machine Learning* to automatically recognize protein name based on features from the *GSearch Engine*. In the third paper, GSearch is substituted with Google, and the task in this paper is to extract all unknown names belonging to one of 273 biomedical entity classes, like genes, proteins, processes etc. After getting promising results with Google, the fourth paper shows that this approach can also be used to retrieve interactions or relationships between the named entities. The fifth paper describes an online implementation of the system, and shows that the method scales well to a larger set of entities.

The final paper concludes the "proof of concept" research, and shows that the performance of the original GeneTUC NLU system has increased from handling 10% of the sentences in a large collection of abstracts in 2001, to 50% in 2006. This is still not good enough to create a commercial system, but it is believed that another 40% performance gain can be achieved by importing more verb templates into GeneTUC, just like nouns were imported during this work. Work has already begun on this, in the form of a local Masters Thesis.

Contents

List of Figures

List of Tables

Preface

This doctoral thesis is submitted to the Norwegian University of Science and Technology (NTNU) in partial fulfillment of the requirements for the degree *Doktor Ingeniør*.

The work herein was performed at and funded by the Department of Computer and Information Science, NTNU, under the supervision of Associate Professor Tore Amble. It is the first result from the new PhD school at NTNU, which allows students to start the PhD study while still writing their Master Thesis.

This thesis consists of three parts. The first part introduces the work and key research fields. The second part is the main contribution, presented as a collection of six research papers in the same format as they were originally published. Finally, the third part concludes the thesis and gives some ideas on future work and an outlook of the field.

Acknowledgments

Firstly, I would like to thank my supervisor Associate Professor Tore Amble. The support from him, together with the co-advisors Astrid Lægreid, Amund Tveit and Arne Halaas, made this work possible. Also, thanks to Anders Andenæs and Tor-Kristian Jenssen who started the GeneTUC project.

I would like to thank Professor Franz Günthner for inviting me to a stay at the Ludwig Maximillian University in Munich and Professor Jun'ichi Tsujii for inviting me to stay at the University of Tokyo. The collaboration with the people working in these two laboratories has been really inspiring, and of great help to my research.

The biologists at the Department of Cancer Research and Molecular Medicine deserves credit for being very supportive and for helping me with all the biology related issues of the work. Astrid Lægreid coordinated the cooperation, and arranged classes like "Medicine for none-practitioners" and "Molecular Biology

for Technologists". Liv Thommesen, Tonje Strømmen Steigedal, Kristine Misund and Kamilla Stunes did a good job evaluating results of the experiments and helping me getting them published.

I would like to thank all the people at the Division of Intelligent Systems at NTNU. Especially, Amund Tveit, Martin Thorsen Ranang and Harald Søvik for their contributions to research included in this thesis and Waclaw Kusnierczyk and Tore Bruland for sharing their ideas with me. Thanks also to Jörg Cassens and Magnus Lie Hetland for invaluable technical advice regarding LaTeX and Python, and Morten Hartmann for teaching me how to use the PhD thesis template.

Thanks to Hallgeir Bergum and Heri Ramampiaro who made a local copy of MEDLINE available at NTNU, and thanks to Bjørn Kåre Alsberg who pointed out business opportunities in this research.

Additionally, I would like to thank my friends, colleagues and *The Lunch Bunch*, including Richard Blake, Peter Hughes, Rolv Inge Seehuus, Christian Mönch, Per Kristian Lehre, Axel Tideman, Frode Sørmo, Mingyang Gu, Xin Tong, Trond Schjelderup Hegdahl, Annette Lykknes, Roland Wittje, Jahnavi Phalkey, Nils Grimsmo, Fredrik Orderud, Clemens Marschner, Holger Bosk, Sebastian Nagel, Mariya Vitusevych, Petra Maier, Yoshimasa Tsuruoka, Jin-Dong Kim, Hong-Woo Chun, Takashi Ninomiya and Yusuke Miyao. Thank you for interesting lunch and dinner discussions almost every day the last five years. You each provided small important pieces of advice, that have guided both my work and my life in different ways.

Nancy Lee Eik-Nes at NTNU taught me a great deal about scientific writing and oral presentation, and my former classmates Frode Høyvik, Jørgen Austvik, Carl-Fredrik Hersoug, Maria Bartnes Dahl, Per Håkon Meland, Bernt Kåre Johannesen and Henrik Tveit deserve credits for motivating me and proof-reading a few of my manuscripts.

In the final stage of the thesis work, I really appreciated the presentations and discussions at the first international symposium on Language in Biology and Medicine (LBM) in Korea. Thanks to Jong C. Park and his good helpers at KAIST, and Limsoon Wong from IIR in Singapore, for arranging it, and for inviting me on the "Invited Speakers Tour" where I appreciated the chance to discuss my ideas with Alfonso Valencia, Nigel Collier, Udo Hahn, Vladimir Bajic and See-Kiong Ng. I am also grateful for stimulating discussions with Jian Su at IIR in Singapore during my multiple short visits the last year.

I would like to thank my family for love and support through three decades, and finally, I would like to thank Nahoko for her support, and for giving me the extra motivation needed to finish this PhD on time.

May 29, 2006, Rune Sætre

Part I

Context

Chapter 1

Introduction

"Can Machines Think?" — Alan Turing (1912-1954)

The main introduction is given in Section 1.1. Section 1.2 states the original task specification and Section 1.3 presents the research questions that have been identified and explored. An outline of the thesis is shown in Section 1.4.

1.1 Introduction

The idea of intelligent machines or Artificial Intelligence (AI) is a very old concept. Many philosophers have asked the question "Can Machines Think?", and since 1950 there have been numerous attempts to make programs that act intelligent according to the Turing Test described below. Earlier it was believed that intelligence was required to win a game of for example chess or bridge, but in 1997 the world chess champion, Gary Kasparov, was beaten by IBM's computer Deep Blue, and still most people do not think Deep Blue is intelligent.

In the field on Computer Science, AI is tightly connected to Natural Language Understanding (NLU). In 1950 the British mathematician Alan Turing made the Turing Test as a way to objectively tell if a machine is intelligent or not ("Can it think?") [17]. In this test, a human *tester* uses natural language to communicate with an *unknown entity* via an electronic chat session. If the *tester* cannot distinguish *a machine* from *another human*, then the machine is judged as intelligent.

In the Natural Language Understanding research community, there has been much focus on understanding news articles, including automatic translation and summary generation. This has been especially well funded by USA after the terror attack in New York, September 11th, 2001. Basically, DARPA wanted a

system that could create a warning or a summary every time a terrorist name or something related was mentioned on the Arabic News Channels or in a tapped phone call. The performance of the translation programs is usually measured automatically with BLEU (Bilingual Evaluation Understudy) scores, where 1.0 is the best [13]. The best systems only achieve around 0.5, but even human translators cannot get a score of 1.0, unless the text matches word by word with one of the alternative gold standards. When it comes to the simpler task of Entity Recognition (ER), e.g. finding company or person names in the text, performance is much higher, and comparable to that of humans solving the same task.

NLU is a part of the Natural Language Processing (NLP) field, which also includes simpler methods that do not use parsing or syntax like NLU. Most of the other NLP systems do not pretend to be *intelligent*, but they are still very *useful*. One of the most useful services in this category is probably Google. Google has a copy of all the web pages on Internet, and in less than a second, they can tell you all the pages that contain the words you are searching for. If you type in a question, there is usually no attempt to answer it, but you can still get the answer, because Google points you to the correct *Question and Answer* page, for example. Or, if you include a question mark in your query to Google, you will get a tip about the Google Answer service, which involves human researchers, and costs money.

Another important event that took place in 2001, was the publishing of the complete Human Genome sequence by the International Human Genome Mapping Consortium [6] and Celera [7]. This led to a surge of research papers in molecular biology and many related fields, and the already fast growing collection of Medical abstracts, MEDLINE, started growing even faster. See Section 2.3.1 for more details on MEDLINE. It also led to a good funding situation for other related research areas, like NLP. The NLP community took this chance to apply the methods from the newspaper domain to the medical domain. Unfortunately, computers did not have nearly the same success rate at recognizing names from medical papers, as they did in news articles. This is why more research in parsing of medical domain articles is needed, and hopefully the answers will help, also in other highly complex domains.

This thesis describes work done within the GeneTUC project. GeneTUC was created to help biologists stay on top of the ever increasing pile of published research that they should know about. GeneTUC is based on "The Understanding Computer" (TUC) system, which was made in the early 1990s from the HSQL system [1]. The contributions are mainly on the semantics side, but the grammar definitions have also been improved a lot during the thesis work. Even though GeneTUC is in the NLU domain, most of the papers in this thesis uses simpler NLP techniques to construct parts of the overall NLU system. The most important subsystem is called Bioogle, because it uses Google to find biomedical

facts.

The GeneTUC parser requires a semantic network (ontology) to produce correct parses, so building the correct semantic network is very important. The network is built (manually) and updated when new types of input text require it. It is usually not possible to update the semantic network based on the input text alone, so in situations where words are missing from the existing network, manual work or programs like the ones described in this thesis are needed. Also, because there is not yet a full consensus in the biological community about what the "correct semantic knowledge" is (i.e. the knowledge and standards are shifting), the ontology has to be flexible. Important standardization work is currently being done, for example with Gene Ontology (GO), GENIA and other ontology building projects. The work in this thesis can be used as an automatic supplement to reduce the manual labor in some of these other resource-creating projects.

1.2 Original Task Specification

> *Natural language processing of gene information:*
> *Current knowledge about genes and their interactions exist largely only as free text. Searching and cross-linking such information rely largely on existing indexes created either manually or by syntactic pattern matching. As a first step we want a tool that is able to correctly recognize occurrences of a gene in free text, e.g. in an article abstract, and the context in which the gene is mentioned. The project will be in cooperation with Biomedical research at NTNU, and will partly be building on existing prototypes for text mining of biomedical texts (GeneTUC).*

The purpose of the GeneTUC project is to find out if full parsing works in the biomedical domain. TUC has already proved that full parsing can be used in simpler domains, like BusTUC in the bus scheduling domain [2]. The purpose of GeneTUC is not to meet several end-user demands, or to create a commercial product, even though BusTUC has showed that this can be a future possibility. It should be shown that the method of full parsing works better than for example the statistical approach used in PubGene [10] or simple lexical matching as in Entrez PubMed[1]. In PubGene a relation connects two genes if there are many text abstracts that mention both of the genes together, but this often leads to many falsely predicted relations. The GeneTUC project should show that the quality of the extracted relations will be better when the extraction is based on full parsing.

[1] www.ncbi.nlm.nih.gov/PubMed/

Because of the computational complexity of full parsing, some pre-processing is needed to parse all the text in a reasonable amount of time. One example of such pre-processing is to extract only the subset of articles that contain certain keywords, and then do full parsing on them. In that case, less parsing is needed before an answer can be found. Such a system will then be a hybrid between plain text searching and strict grammar parsing. In GeneTUC, this approach has been simulated by focusing mostly on a subset of MEDLINE, namely all abstracts containing the word "gastrin" (more than 12.000).

The long term goal for GeneTUC is to successfully parse all grammatically correct sentences in any given abstract from MEDLINE. This is a difficult problem, because in addition to all the usual NLP challenges, like synonyms, homonyms, ambiguities etc, there is also the challenge that biologists use a more complex syntax in this kind of compact text. Still, TUC is relying on full parsing, so this is a necessary goal for GeneTUC too, as long as it is being developed using the TUC architecture.

A follow-up goal is to make use of the semantics in the parsed sentences. GeneTUC should by default be able to answer questions like *"What represses PKC expression?"*, if that fact is stated somewhere in the input abstract. When GeneTUC can answer these kinds of questions, the next step is to evaluate how useful the system is to the biomedical society. That means that GeneTUC will have to solve real problems, posed by practicing biologists. The last step before making a commercial system is to make sure that the system is as good as or better than other existing approaches. In other words, GeneTUC should be comparable to other systems that use statistical approaches, machine learning or other techniques to get similar results. These tasks are summed up as research questions in the next section.

A sub-goal is to keep the same grammar for all the TUC applications, to maintain sideways compatibility with for example BusTUC[2]. BusTUC was launched as a commercial on-line application in 1997, and it is able to answer 98% of all well-formed questions about bus departure and prices in Trondheim city correctly. The input can be given in either English or Norwegian, and the system fixes spelling errors in place names on the fly. The average response time of BusTUC is around 1 second per question.

1.3 Research Questions

The main research question identified and explored by this thesis is:

Can computer programs like GeneTUC understand text about molecular

[2]http://www.idi.ntnu.no/~tagore/bustuc/

biology as well as humans can?

A detailed breakdown of the main research question is given here:

1. Is our approach possible?
   (a) *Can full parsing be used to extract information from these texts?*
   (b) *Can new search technologies, like Google API*[3]*, be used in this work?*
2. Is our approach useful?
   (a) *How many percent of the sentences can be understood by a computer?*
   (b) *How fast are computers at* understanding*, compared to humans?*
3. Is our approach better than others?
   (a) *Is Bioogle a better way to build dictionaries?*
   (b) *Is Bioogle a better way to build ontologies?*
   (c) *Is GeneTUC a better way to do Information Extraction in Medical Text?*

1.4 Thesis Outline

The thesis is a *collection of papers*. The outline is as follows:

Part I - Context

Chapter 1 gives the introduction to the thesis.

Chapter 2 presents the relevant scientific fields.

Chapter 3 summarizes the research contributions and the motivation behind them. Also, a complete publication list is given.

Part II - Papers

This part presents the main research contributions of the thesis. A collection of six research papers is given, with the papers being in the same format as they were originally published.

[3] Application Programming Interface

Part III - Synopsis

Chapter 4 contains an errata for the published papers. To avoid further confusion, the mistakes that have been discovered after the papers were published, are all listed here.

Chapter 5 concludes the thesis, suggests possible avenues for future work and gives a short outlook on the challenges facing the research field.

Chapter 2

Background

> "Computers are not intelligent. They only think they are."
> — Anonymous

> "The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation."
> — Tim Berners-Lee (1955-)

In 1950 the British mathematician Alan Turing concluded that the question "Can Machines Think?" has ambiguous meaning, so he offered his Turing Test as a less ambiguous substitute [17]. In his test, a human tester communicates via an electronic chat session with an unknown entity, either a machine or another human. If the tester cannot distinguish the machine from another human, then the machine is judged as intelligent. During the past 50 years there have been numerous efforts to write programs that fool testers into believing that they are human. Some of these programs, without being truly intelligent, have been successful against some testers [9].

This chapter presents the main relevant scientific fields that intersect the NLU research in this thesis. Section 2.1 introduces Information Retrieval (IR) in general, and then the more specialized task of Information Extraction (IE). Section 2.2 introduces the general notion of Natural Language Processing (NLP), and the more specialized notion of Natural Language Understanding (NLU), which is in the intersection between the IE and the NLP research fields (see Figure 2.1). Section 2.3 gives a brief introduction to related research in the BioNLU domain.

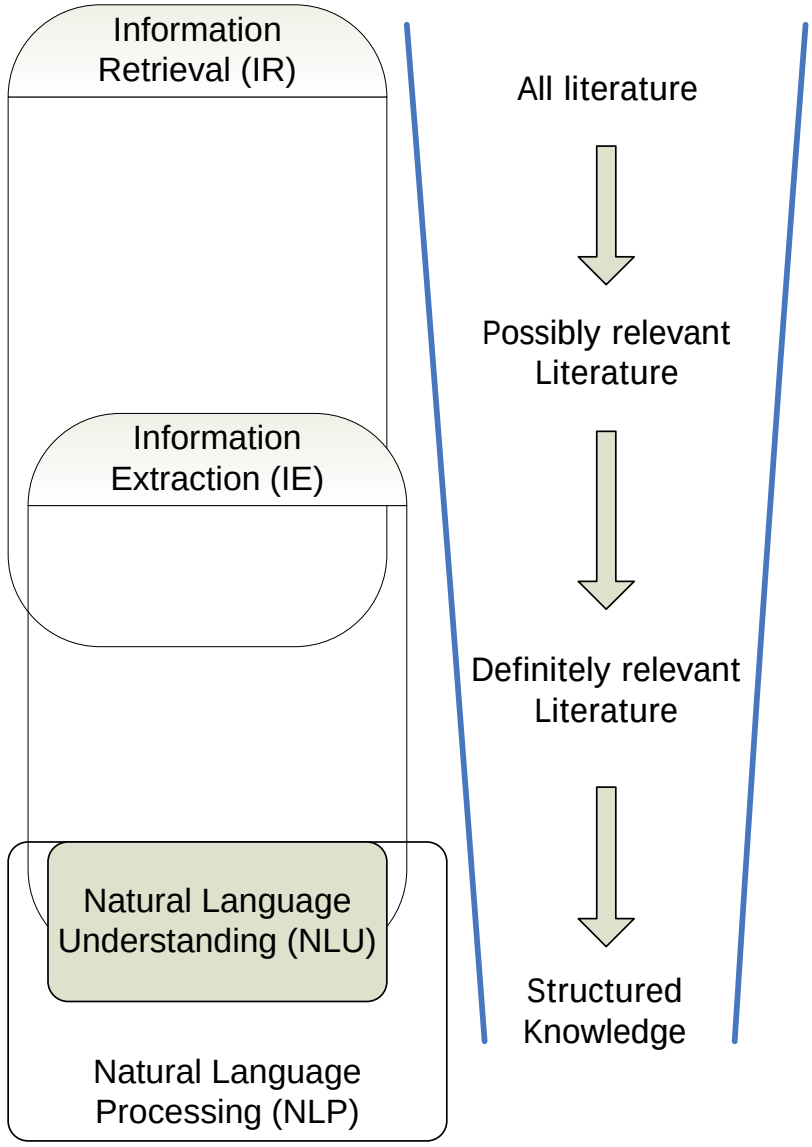

Figure 2.1: Relationship among IR, IE, NLP and NLU, and the "relevance funnel"

2.1 Information Retrieval and Extraction

Information Retrieval (IR) is the task of retrieving documents to help a user find the right answer. The simplest form of modern IR is the use of Uniform

Resource Locator (URL) in the World Wide Web. The user inputs a URL to a web-browser, and then gets the corresponding web document in return. This gets a little more advanced with the use of hyperlinks between documents, since the user then can retrieve more relevant information by following the links from one document to other similar or complementary documents. More advanced systems, like Google, ask the user to provide a query with some words that are expected to occur in the relevant documents, and then the system tries to create a prioritized list where ten of the most relevant suggestions are on the first page. This is in the borderline between IR and Information Extraction (IE). This can be seen in the funnel in Figure 2.1, where useless information is (hopefully) filtered away towards the bottom of the funnel. At the bottom of this funnel, there are no longer entire documents from the original literature, instead there are extracted facts that are assumed to be of interest to the user.

2.2 Natural Language Processing & Understanding

There are many definitions of NLP available. The one presented here is used at the University of Birmingham[1], and clarifies what is meant by Natural Language in this context.

Definition 1 *A 'natural language' (NL) is any of the languages naturally used by humans, i.e. not an artificial or man-made language such as a programming language. 'Natural Language Processing' (NLP) is a convenient description for all attempts to use computers to process natural language. NLP is often used in a way which excludes speech, and then SNLP is used as the term to include both speech and other aspects of natural language processing.*

NLP includes Speech synthesis, Speech recognition, Natural Language Understanding (NLU), Natural language generation and Machine Translation (MT).

The GeneTUC Question Answering system is not concerned with Speech, since the input is expected to come from literature databases, or from a keyboard, and the output is not spoken, but written to a computer screen. There is also no MT involved since the working language is expected to be English for all researchers.

There are both NLU and simple Natural Language Generation (NLG) in GeneTUC. NLU is needed to move from isolated words in the text to 'meaning', and involves the ontological model systems and the grammar. Some simple NLG is used to generate appropriate NL responses to the unpredictable input from a user, but the main focus of this thesis work is on NLU on the input side.

[1] http://www.cs.bham.ac.uk/~pxc/nlpa/2002/AI-HO-IntroNLP.html

NLU is a part of the more general fields of Information Extraction (IE) and Natural Language Processing (NLP), which were described in the previous sections and can be seen in Figure 2.1. The history of NLU can be traced back to the logic-based paradigm that was begun by the work of Colmerauer and his colleagues on Q-systems and metamorphosis grammars in 1970. They were the forerunners of Prolog, and Definite Clause Grammars, which is what the GeneTUC system is built on top of. Other research from the same time period also inspired TUC, including functional grammar, and Lexical Functional Grammar (LFG).

The NLP researchers Jurafsky and Martin describes the birth of NLU like this in their widely used text book on *Speech and Language Processing* [12]: "The natural language understanding field took off during the 1970s, beginning with Terry Winograd's SHRDLU system, which simulated a robot embedded in a world of toy blocks. The program was able to accept natural language text commands (Move the red block on top of the smaller green one) of a hitherto unseen complexity and sophistication. His system was also the first to attempt to build an extensive (for the time) grammar of English, based on Halliday's systemic grammar. Winograd's model made it clear that the problem of parsing was well-enough understood to begin to focus on semantics and discourse models. Roger Schank and his colleagues and students (in what was often referred to as the *Yale School* built a series of language understanding programs that focused on human conceptual knowledge such as scripts, plans and goals, and human memory organization. This work often used network-based semantics and began to incorporate Fillmore's notion of case roles into their representations."

The logic-based and natural-language understanding paradigms were unified on systems that used predicate logic as a semantic representation, such as the LUNAR question-answering system. Question Answering (QA) is an important part of NLU, and has been researched since the 1960s. Examples of modern QA systems are AskJeeves[2] which has been around since the dawn of the World Wide Web, but never became very popular, and AnswerBus which was first presented at the Human Language Technology conference in 2002, and scored a 70% precision score on the TREC-8's 200 test set [20]. Having common challenge tasks like the Text REtrieval Conference (TREC[3]) and the Message Understanding Conference (MUC[4]) is very important when it comes to choosing the best approach to building useful systems. These challenges and conferences highlight the strengths and weaknesses of the different systems, and guide the researchers in choosing which approaches to build further on.

GeneTUC is a QA system based on predicate logic as its semantic representation, just as the LUNAR system was. The biggest challenge that GeneTUC faces

[2] askjeeves.com
[3] http://trec.nist.gov/
[4] http://www.itl.nist.gov/iaui/894.02/related_projects/muc/

is because of the chosen domain of BioNLU. While LUNAR worked in a closed domain, just answering questions about lunar geology and chemistry, with a fixed set of English words [18], GeneTUC is working in the much broader domain of molecular biology. The challenges include use of complex names, invention of new names every day, and use of generally complex *research language* in the abstracts and articles that should be parsed.

2.3 NLP in Medical Text

The Entrez PubMed[5] approach to searching the MEDLINE database falls under the definition of Information Retrieval (IR). The simplest form of IR is to enter words or protein names into the search engine, and then a list of all articles or abstracts containing the keywords are returned. Entrez also offers slightly more advanced IR, where the input can be an amino acid sequence for example, and the results are records of well-known proteins that contain this sequence.

GeneTUC uses more advanced techniques from NLU and IE research, adding some AI extensions to the simple IR task. In this case, it means that a full parser parses all the input sentences and transforms each sentence into some logical formula that represents the meaning of the original sentence. This transformation is called text mining. Then the researchers can state their questions using natural language sentences, and the system should be able to find the right answer to the question, or at least give a list of only the abstracts/articles that are semantically relevant to the question.

In this book the term "BioNLU" will be used as a short form for "Natural Language Understanding research in Molecular Biology Literature". Papers I and VI describe the BioNLU methods that were used and the main results that were obtained in the GeneTUC project.

In order to work with GeneTUC, knowledge of the field of molecular biology is crucial in addition to traditional linguistic knowledge. This is to avoid being blinded by the sometimes seemingly incomprehensible sentences written in this kind of molecular biology research literature. The book *Biology* by Raven et al. [14] provides an excellent and thorough introduction to many of the necessary topics in the field.

Judging from the participation in the MUC and TREC conferences, there are more than 40 groups in the world trying to extract protein-to-protein interactions from abstracts using NLP. Two approaches that have many similarities to GeneTUC are "Event extraction from biomedical papers using a full parser" [19], and "BioIE: Retargetable Information Extraction and Ontological Annota-

[5] http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

tion of Biological Interactions from the Literature" [11]. In these systems, full parsing and many hand-made rules are used, and this leads to very good precision scores. In the first paper, 31 events were uniquely extracted, and 32 events had two or more ambiguous argument structures. This gives a precision score around 33%. 63 of 133 events were extracted, giving a recall score around 50%. The combined F-measure was around 40, but it was expected to rise towards 74, using post-processing of ambiguous and partial results. An improved version of this grammar was used when all the MEDLINE abstracts were parsed by the Tsujii-lab in November 2005. In the second paper, recall and precision were 55%∼57% and 88%∼92% respectively, giving an F-measure of 68∼70. In this experiment, 600 sentences were used, and the test corpus is now available online. Both of these papers discuss the balance between precision and recall scores, and point out that in certain precision sensitive applications it is necessary to increase the precision at the cost of a lower recall level.

The semantic network in TUC needs to be upgraded with more molecular biology terms, and there are several places to look for these new words. One possible source is WordNet[6], which defines *word contexts* and semantically connected *clusters of lexical words (meanings)*. A more scientifically relevant source to GeneTUC is Gene Ontology, which defines molecular functions, biological processes and cellular compartments that can be coupled with specific proteins through an annotation process [8]. Part of the GeneTUC project is to incorporate words from these sources into TUC. Earlier, all adverbs and adjectives have been imported into TUC from WordNet. All verbs were not directly imported from WordNet in the same manner, since it is believed that the group of relevant verbs is small enough to allow for manual insertion, which gives better precision than automatic definitions. Some of the *new words* may be defined only in the abstract currently being parsed. These words must then be added *on the fly* to the GeneTUC system.

The most obvious way to add new terms to the system is by manual labor, preferably by biologists, but this usually takes too much time. The second best option is to use already established dictionaries, but most of the time the coverage of the dictionary is not good enough. Papers I, II and III focuses on different ways to do automatic recognition of *unknown terms*, using online web resources.

2.3.1 PubMed MEDLINE

The best available source of molecular biology paper abstracts is the MEDLINE Databases, which is maintained by the American association National Center for Biotechnology Information (NCBI). NCBI was established in 1988 as a national (American) resource for molecular biology information, but their resources are

[6] http://www.cogsci.princeton.edu/cgi-bin/webwn1.7.1

now widely used throughout the world. NCBI creates public databases, conducts research in computational biology, develops software tools for analyzing genome data, and disseminates biomedical information - all for the better understanding of molecular processes affecting human health and disease.

The growth in PubMed is the primary motivation for building systems like GeneTUC. Figure2.2 shows that, on average, nearly two thousand new abstracts are published and indexed in PubMed every day now, so computers are needed to help the researchers deal with this information overload.

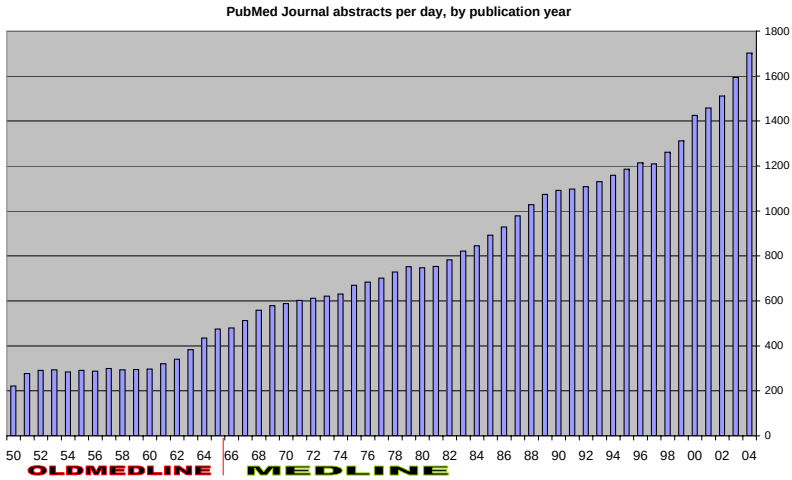

Figure 2.2: MEDLINE Growth

Chapter 3

Research Summary

> "The characteristic of scientific progress is our knowing that we did not know." — *Gaston Bachelard (French philosopher and poet, 1884-1962)*

> "If it works, it's not AI" — *Computer science joke from the 1980s*

This chapter will give a summary of the research documented in this thesis. Firstly, Section 3.1 describes the research process behind the contributions with a focus on motivational elements and major choices that were made. The paper abstracts are given in Section 3.2, and a specification of the author's contributions to each of the papers are given in Section 3.3. Finally, the author's publications are listed in Section 3.4.

3.1 Research Process

This section will give a chronological description of the research process behind the research contributions. The emphasis here is on the motivation behind the different contributions and the major choices that had to be made in the process. More specific information and technical details of the contributions can be found in Part II where the contributions are presented as self-contained research papers.

A conceptual illustration of the research process with references to the relevant contributions is shown in Figure 3.1. This section and its subsections follows the flow depicted in the figure.

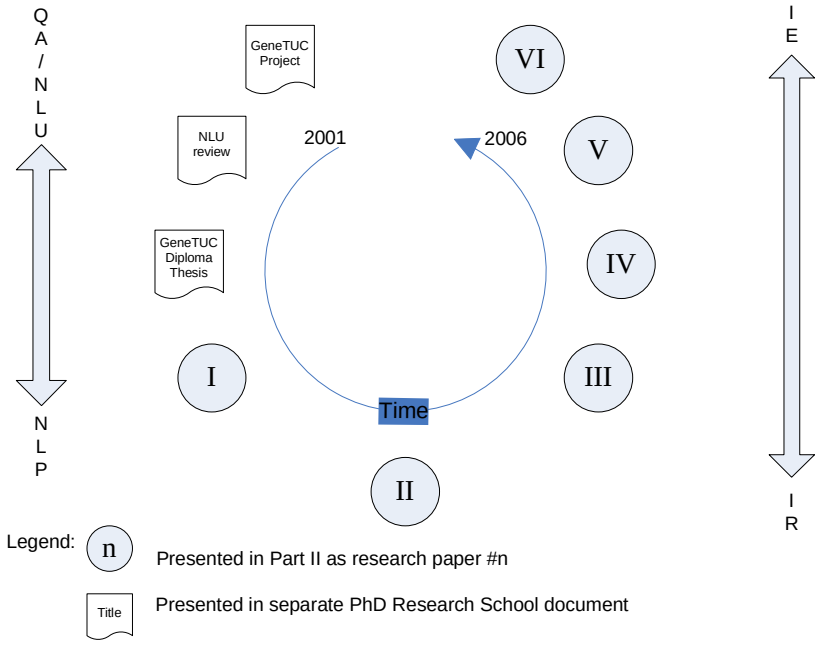

Figure 3.1: A conceptual illustration of the research process and relevant contributions

3.1.1 Initialization

The GeneTUC research project was initiated in the year 2000, which is also the year that the first complete sequence of human genome was put together. One year later, the PhD research school at NTNU was started, giving master students a chance to combine their Masters thesis and PhD thesis into one coherent work. The author of this thesis was one of the three first candidates to start in this program in August 2001, and is now finishing after five years of combined Masters and PhD work on GeneTUC. GeneTUC was originally started by Anders Andenæs [3, 4] and taken over by the author of this thesis in 2001. A detailed description and the previous history of the system is presented in early project reports [15, 16].

3.1.2 Local Grammars

The first research contribution in this thesis is an overview of the BioNLU field, and an experiment that uses Local Grammars to extract protein names and interactions from molecular biology texts. Paper I includes the review and a plan for how the GeneTUC system should be improved. Since it was written while the author was visiting the Center for Information and Language sciences (CIS) at Ludwig-Maximillian University in Munich, it also includes an example of how local grammars can be used in the information extraction process. The local grammars were part of the inspiration behind the idea of using very simple queries in the Bioogle systems mentioned below.

3.1.3 Machine Learning

The second paper attacks the problem of identifying protein names in a more automatic manner than Paper I, namely by using Machine Learning. After Paper I, it was obvious that the amount of manual labor needed in the GeneTUC and other similar knowledge intensive systems is huge, and more automatic approaches have to be sought for. The first attempt is presented in Paper II. It uses the search engine *GSearch* to create the features needed by the machine learning algorithms. GSearch has an Application Programming Interface (API) to the local NTNU copy of the most famous protein databases in the world.

3.1.4 Google

During the writing of Paper II, it became obvious that APIs to public search engines have great potential in helping identify named entities. At the same time, Google released a similar API to their search engine, and that could also be used to discover interactions between the entities from online literature. Papers III and IV describe experiments with the Google API as the main source for this kind of knowledge. Together with Paper V, this constitutes the core contribution of the PhD Thesis, all based on the same main idea of using few, simple queries.

Paper III was a milestone in that it showed that Google can be used to simplify the information discovery process in a more human understandable way than what was presented in Paper II. Paper IV, carried the work into a more complex relationship extraction research, and in Paper V, the findings from Papers III and IV were put into an online system, making the workflow between biologists and computer scientists easier. Also, more experiments were done to verify that the results would hold across more protein names.

3.1.5 GeneTUC and Future Technologies

The last paper, Paper VI, presents a summary of the entire GeneTUC project so far, and explains how all the new pieces from the other papers can be put into the main system. Also, performance of the system is measured and compared to other systems. This final research contribution shows that the GeneTUC system performs 40% better than before, mainly because all the unknown entities from the benchmark text have been added.

A lot of changes were made in the grammar during this project. The general TUC grammar is much better now than it was before the GeneTUC project started, but much work can still be done to improve the recall of the system further. This must be done in such a way that the constructs that have already been added are not destroyed in the process. Therefore it is important to have the kind of standards for measuring the performance that were introduced in Paper VI.

As seen in Figure 3.1, the last paper (VI) completes the research circle. It started out from a prototype Question Answering (QA) system in 2001, went through stages of more and more detailed, and smaller domain, problem solving, before moving back towards the high-level goal of building a complete system in the last papers. Even though the circle is now completed, one or two rounds are still needed, before the GeneTUC application can be truly useful for the intended end users. In the next loop through the circle, the focus should be on events, which are usually represented by verbs, in the text. Throughout the thesis work, the parsing performance of the system has increased from 10% to 50%. It is believed that another 40 percent points can be achieved by improving the (semantic) coverage of verbs in GeneTUC, together with continuous improvements in the general grammar.

3.2 Paper Abstracts

3.2.1 Paper I

GeneTUC: Natural Language Understanding.
Automatic Information Extraction from Biomedical Texts.

The article is about Natural Language Understanding (NLU) in general and more concretely about its application to molecular biology texts on the topic "gene- and protein-activations". The first part is a short review of different current research approaches in the field of NLU and "bio-linguistics". The second part look into the bottom-up grammar building approach that is sketched in the article "The Construction of Local Grammars" by Maurice Gross. The visualization system "Unitex", made by Sébastien Paumier, is used to construct the local grammars. The results will be compared to the full-parsing approach used in GeneTUC. The goal is to integrate Unitex with GeneTUC in a way that will get the best from both worlds of shallow and deep parsing. The preliminary results suggest that the medical language is constrained enough for the Local Grammar approach to work. 38 graphs were constructed to capture the essence of 59 "activate-sentences", and 18 graphs where created to capture all the different entity names that were used in the sentences. When the graphs were applied to a new text for testing, many of the constructed "activation-patterns" also matched in the new text.

3.2.2 Paper II

ProtChew: Automatic Extraction of Protein Names from Biomedical Literature.

With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. Due to incomplete biomedical information databases, the extraction is not straightforward using dictionaries, and several approaches using contextual rules and machine learning have previously been proposed. Our work is inspired by the previous approaches, but is novel in the sense that it is fully automatic and does not rely on expert tagged corpora. The main ideas are 1) unigram tagging of corpora using known protein names for training examples for the protein name extraction classifier and 2) tight positive and negative examples by having protein-related words as negative examples and protein names/synonyms as positive examples. We present preliminary results on MEDLINE abstracts about gastrin, further work will be on testing the approach on BioCreative benchmark data sets.

3.2.3 Paper III

Semantic Annotation of Biomedical Literature Using Google.

With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. Due to incomplete biomedical information databases, the extraction is not straightforward using dictionaries, and several approaches using contextual rules and machine learning have previously been proposed. Our work is inspired by the previous approaches, but is novel in the sense that it is using Google for semantic annotation of the biomedical words. The semantic annotation accuracy obtained - 52% on words not found in the Brown Corpus, Swiss-Prot or LocusLink (accessed using Gsearch.org) - is justifying further work in this direction.

3.2.4 Paper IV

gProt: Annotating Protein Interactions Using Google and Gene Ontology. With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. Due to incomplete biomedical information databases, the extraction cannot be done straightforward using dictionaries, so several approaches using contextual rules and machine learning have previously been proposed. Our work is inspired by the previous approaches, but is novel in the sense that it combines Google and Gene Ontology (GO) for annotating protein interactions. We got promising empirical results - 57.5terms as valid GO annotations, and 16.9provided by our system gProt. The total error-rate was 25.6mainly of overly general answers and syntactic errors, but also including semantic errors, other biological entities (than proteins and GO-terms) and false information sources.

3.2.5 Paper V

WebProt:
Online Mining and Annotation of Biomedical Literature using Google.

Motivation: WebProt is an open source software tool for text mining in molecular biology texts. It is used to collect background information about genes and proteins from online literature sources. This is useful for molecular biologists working with many unfamiliar genes, like for example in a big microarray experiment.
Results: A study using 42 different proteins and their official synonyms/aliases showed that 69,5% of all the results from Google were useful, i.e. containing either *related protein names* or *biological functions, locations and processes* related to the query protein. This is 10% lower than in a previous experiment, so we made

a filter requiring each web page to yield at least 6 results, before being taken into consideration. This increased the precision to 91,5%, but the recall dropped to $\frac{2}{3}$, thereby lowering the F-measure of the system from 82% to 78%.
Availability: WebProt is available as a web service system running on http://www.idi.ntnu.no/~satre/webprot/.
It can also be freely downloaded for academic purposes by sending a request to the first author of this chapter. The advantage of using the web based system is that you get access to all the results submitted and searched for by other biologists, and this will speed up some of your searches tremendously.

3.2.6 Paper VI

GeneTUC, GENIA and Google:
Natural Language Understanding in Molecular Biology Literature.

With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. GeneTUC has been developed to be able to read biological texts and answer questions about them afterwards. The knowledge base of the system is constructed by parsing MEDLINE abstracts or other online text strings retrieved by the Google API. When the system encounters a word that is not in the dictionary, the Google API can be used to automatically determine the semantic class of the word and add it to the dictionary. The performance of the GeneTUC parser was tested and compared to the manually tagged GENIA corpus with EvalB, giving bracketing precision and recall scores of 70,6% and 53,9% respectively. GeneTUC was able to parse 60,2% of the sentences, and the POS-tagging accuracy was 86.0%. This is not as high as the best taggers and parsers available, but GeneTUC is also capable of doing deep reasoning, like anaphora resolution and question answering, and this is not a part of the other parsers.

3.3 Authors' Contributions

Here is a list of the authors' contributions to each of the papers. The main author with his contributions is listed on the first line, and then the others with their contributions, in descending order, on the following lines. See Table 3.1 for a list of each author's acronym.

3.3.1 Paper I

RS Literature review, experiments and writing the paper.

AT	Amund Tveit	Bio	Biologists
HS	Harald Søvik	AL	Astrid Lægreid
MTR	Martin Thorsen Ranang	KS	Kamilla Stunes
RS	Rune Sætre	KM	Kristine Misund
TA	Tore Amble	LT	Liv Thommesen
YT	Yoshimasa Tsuruoka	TSS	Tonje Strømmen Steigedal

Table 3.1: A list of contributing authors and their acronyms, sorted alphabetically and split in groups of computer scientists and biologists.

3.3.2 Paper II

AT The machine learning idea, step 7 and 8 (see Figure 1 in the paper), made the figures, wrote Sections 1 and 4, supervised the research and edited the paper.

RS Steps 1,2,3,4,5,6 and 10 (see Figure 1 in the paper), implemented the NLP parts, made the interface to GSearch, selected the features for machine learning, coordinated the expert evaluation, and wrote Sections 2, 3, 5, 6 and 7 of the paper.

Bio AL and TSS:
Evaluated the biological results, under the supervision of AL.

3.3.3 Paper III

RS The idea about using Google API for entity extraction, implemented the system and wrote sections 2, 3, 4, 6 and 7 of the paper.

AT Wrote the Abstract and sections 1 and 5, made the figures, and supervised the research.

Bio TSS and AL:
Evaluated the biological results, under supervision of AL.

3.3.4 Paper IV

RS The idea of using Google API to find protein interactions and GO evidence text from the Web, implemented steps 1, 2 and 3 (see Fig. 1 in the paper), and wrote sections 2, 3, 4 and 6, and edited the paper.

AT Wrote sections 1, 5, 7 and Future Work, made the figures and supervised the research.

MTR Implemented step 4 (see Fig. 1 in the paper): a quick index matcher for Gene Ontology terms.

Bio TSS, LT, KS and AL:
Evaluated the biological results, under the supervision of AL.

3.3.5 Paper V

RS Implemented the system and wrote the paper (except Subsection 5.3.5).

MTR Implemented and wrote Subsection 5.3.5

Bio TSS, KS, KM, LT and AL:
Tested the system and evaluated the biological results, under the supervision of LT.

3.3.6 Paper VI

RS Implemented the system, made figures and wrote the paper (except Subsection 4.5), in addition to coordinating the research.

HS Implemented the EvalB comparison part and wrote Subsection 4.5.

TA Provided input to the TUC-grammar part of the paper.

YT Provided easy access to the GENIA resources, and valuable input to the Related Work section.

3.4 Publication List

Journal Publication

Rune Sætre, Harald Søvik, Tore Amble and Yoshimasa Tsuruoka. GeneTUC, GENIA and Google: Natural Language Understanding in Molecular Biology Literature. In *Special Issue on "Data Mining and Bioinformatics" of Transactions on Computational Systems Biology*, Lecture Notes in BIology (LNBI), Volume 4070, part V, pages 68–82. Editor-in-Chief: Dr. Corrado Priami, University of Trento, Italy. Springer-Verlag Berlin Heidelberg, Germany, 2006. ISSN: 0302-9743.

Book Chapter

Rune Sætre, Martin T. Ranang, Tonje S. Steigedal, Kamilla Stunes, Kristine Misund, Liv Thommesen and Astrid Lægreid. WebProt: Online Mining and Annotation of Biomedical Literature Using Google. Submitted to *Advanced Computational Methods for Biocomputing and Bioimaging*. Editors: Tuan D. Pham, Hong Yan, Denis I. Crane. Nova Science Publishers. Hauppauge, New York 11788-3619. 2006.

International Reviewed Conferences

Rune Sætre, Amund Tveit, Martin T. Ranang, Tonje S. Steigedal, Liv Thommesen, Kamilla Stunes and Astrid Lægreid. gProt: Annotating Protein Interactions Using Google and Gene Ontology. In *Proc. Knowledge-Based Intelligent Information and Engineering Systems (KES) 2005*. Melbourne, Australia, September 14-16, 2005. Lecture Notes in Artificial Intelligence (LNAI) 2005. Volume 3683, Part III, pages 1195–1203. Springer-Verlag GmbH. ISSN: 0302-9743. ISBN: 3-540-28896-1.

International Reviewed Workshops

Amund Tveit, Rune Sætre, Astrid Lægreid and Tonje S. Steigedal. ProtChew: Automatic Extraction of Protein Names from Biomedical Literature. In *ICDE Proc. 21st International Conference on Data Engineering (ICDE'05)*. Tokyo, Japan, April 3-4, 2005. IEEE Electronic Proceedings, pages 1161–1161.
DOI: http://doi.ieeecomputersociety.org/10.1109/ICDE.2005.268

Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal and Astrid Lægreid. Semantic Annotation of Biomedical Literature Using Google. In *Proc. Data Mining and Bioinformatics (DMBIO) 2005*. Singapore, May 9-12, 2005. Lecture Notes in Computer Science (LNCS) 2005. Volume 3482, Part III, pages 327–337. Springer-Verlag GmbH. ISSN: 0302-9743. ISBN: 3-540-25862-0.

Locally Reviewed Conference

Rune Sætre. GeneTUC: Natural Language Understanding. Automatic Information Extraction from Biomedical Texts. In *Proc. Computer Science Graduate Students Conference 2004 (CSGSC-2004)*. Norwegian University of Science and Technology (NTNU). Trondheim, Norway, April 29th, 2004. Electronic Proceedings. URL: http://csgsc.idi.ntnu.no/

Part II

Papers

Paper I

GeneTUC: Natural Language Understanding.
Automatic Information Extraction from Biomedical Texts.
Rune Sætre.
In *Proc. Computer Science Graduate Students Conference 2004 (CSGSC-2004)*
Norwegian University of Science and Technology (NTNU).
Trondheim, Norway, April 29th, 2004.
Electronic Proceedings.
URL: http://csgsc.idi.ntnu.no/2004/

GeneTUC: Natural Language Understanding. Automatic Information Extraction from Biomedical Texts

Rune Sætre
Computer and Information Science (IDI),
Norwegian University of Science and Technology (NTNU)
NO-7491 Trondheim, NORWAY

Abstract

The article is about Natural Language Understanding (NLU) in general and more concretely about its application to microbiological texts on the topic "gene- and protein-activations". The first part is a short review of different current research approaches in the field of NLU and "bio-linguistics". The second part look into the bottom-up grammar building approach that is sketched in the article "The Construction of Local Grammars" by Maurice Gross. The visualization system "Unitex", made by Sébastien Paumier, is used to construct the local grammars. The results will be compared to the full-parsing approach used in GeneTUC. The goal is to integrate Unitex with GeneTUC in a way that will get the best from both worlds of shallow and deep parsing. The preliminary results suggest that the medical language is constrained enough for the Local Grammar approach to work. 38 graphs were constructed to capture the essence of 59 "activate-sentences", and 18 graphs where created to capture all the different entity names that were used in the sentences. When the graphs were applied to a new text for testing, many of the constructed "activation-patterns" also matched in the new text.

1. Introduction

The article contains two main parts. Part 1 is a literature review with the goal of determining where the current research barriers in Natural Language Understanding (NLU) and in computational linguistics applied to micro-biological texts are. Part 2 describes the Unitex & GeneTUC project part of the work.

GeneTUC and Unitex use two different approaches to solve the parsing problem, and GeneTUC could benefit greatly from using parts of the Unitex system for pre-parsing. Especially, the use of graphs when constructing grammars is very promising, and makes it easy also for people with little computer knowledge to produce grammars to fit their needs. How the two systems should best be integrated is still an open question.

2. Previous Work

Following is an overview of existing literature and methods used in the field of Natural Language Processing (NLP) with a special focus on NLU and Information Extraction (IE). The domain for GeneTUC's IE will be biomedical texts describing gene and protein interactions. 15 NLP articles have been reviewed, and together they cover most of the recent advances in biomedical IE. The idea in GeneTUC is that the text is to be fully parsed. Only one other article has been found that describes full parsing of biomedical texts [15], and that article therefore received extra attention The other articles in the collection use various methods of shallow (partial) parsing, or stochastic (statistical) calculations to analyze the language. One of the 15 articles is an interesting article on Local Grammars [9]. It describes the use of Finite State Transducers (Pattern Matching) to extract exact knowledge from texts. This represents a bottom-up approach that will be compared to the top-down approach used in GeneTUC.

2.1 Terminology

There are two current main approaches to information extraction from biomedical texts. NLU is the rule-based, symbolic and grammatical approach, and the other more statistical or pattern matching approach is usually called Natural Language Processing (NLP). Symbolic approaches means using symbols that have a defined meaning both for humans and machines.

This use of terms tells us that NLU seeks to do something more than just process the text from one format to another. The end goal is to transform the text into something that computers can "understand". That means that the computer should be able to answer natural language (e.g. English) questions about the text, and also be able to reason about facts from different texts. The field of NLU is strongly connected to the field of Artificial Intelligence (AI).

Regardless of whether a symbolic or sub-symbolic approach is being used, there is a distinction between full and partial parsing. Full parsing means that every sentence must be completely analyzed from the beginning to the end. The output from full parsing is usually a parse tree saying what Part-Of-Speech (POS) each word has, how words are connected to one another in phrases, and how the phrases together make up the entire sentence. Quite

often there will be more than one possible legal parse tree, and then the sentence must be disambiguated (possibly in a larger context) to find the one intended parse tree (with the right semantics). Another possibility is to simply list all legal parse trees without considering semantics. Partial parsing, on the other hand, means that the output is not a complete parse tree for the entire sentence. Instead it can be smaller parse trees for specific phrases that are recognized in the sentence, or simply a POS-tag for each word, saying nothing about how they connect to each other.

The difference between local and global grammars is somewhat similar to the difference between partial and full parsing. With a global grammar, the dependencies between words far away from each other are modelled explicitly with complex high-level grammatical rules. In the local grammar approach [9], pattern-recognizing automata are built to deal with neighbouring word dependencies. Later these automata can be group into larger units and thereby implicitly solve the long range constraints.

Another criterion under which a parser is evaluated is whether it is robust or not. Robust in this sense means if the parser is able to deal with all reasonable inputs. All parsers are constructed with specific sentence constructions and words in mind, or they are trained (statistically) on a corpus of relevant and already correctly parsed/annotated sentences. However, the human language is so flexible that new and previously unseen constructs or names are bound to appear all the time. When a parser is able to deal in some intelligent manner also with all the examples that it was not specifically constructed or trained for, it is called a robust parser. Most full parsers are not robust, since they are built on the premises that all possible sentence constructs must be known in advance.

In both NLP and NLU many researchers are now trying different corpus-based approaches. That means that they take some collection of actual texts from the domain (e.g. Medline) as a starting point. Then, this text must be manually analyzed by experts in the domain (e.g. Biologists), and tagged by linguistic experts. This pre-processed text can then act as source for learning rules etc., or it can be used as a golden standard when testing parsers, saying exactly what the desired results are for this specific collection of texts.

Future systems are likely to be hybrid systems, including techniques from both of these approaches, since NLU and NLP often offer complementary solutions to the same problem. Sometimes NLU is thought of as a subset of NLP, since "understanding" is also really just some kind of processing. The way GeneTUC understands a text is by translating it into an event-logic form called TUC Query Language (TQL).

One of the goals in the GeneTUC project [14] is to do full parsing of microbiological texts, which is closer to Information Extraction (IE) than to Information Retrieval (IR). While IR and IE are both dealing with some form of text searching, they are quite different in terms of what output or results they produce. IR is the simple classical approach to text searching, as it is done e.g. in Google [18] and other search engines on the Internet. In IR the user enters some words of interest, and then all the documents containing these words are listed. The document list can be ordered accordingly to how many times each search word occurs, how close the different search words are clustered in the document and so on. In this approach, the user has to run many different searches to cover all the possible different search words to describe the fact that she is actually looking for. Also, for every search she might have to read all the articles returned by the search engine, just to see if they really are of interest or not.

IE seeks to reduce the user's workload by adding reasoning to the IR process. With IE the computer will have some knowledge about synonyms and different sentence forms that actually express the same basic facts. That means that the user only has to specify the question that she has, and then the computer will do the tedious work of running several different IR searches, and skimming every single retrieved article to see whether or not it is of interest. The end result from IE can be simple yes/no answers to different questions or it can be specific facts that are extracted from various articles and then used to build databases for quick and easy lookup later.

2.2 Full Parsing

Searching Medline [25], Cite-seer [16] and Google [18], only one article that describes full-parsing of microbiological texts was found. This article is discussed here. Yakushiji et al. [15] is an early report on an experiment that the authors carried out to see if this approach can be used, even when the texts are more complex than e.g. newspaper texts. Their long term goal is to build an information extraction system that can extract specific facts from Medline abstracts. Their short-term experimental goal was to automatically extract 133 (already known) facts from 97 manually annotated test sentences.

The reason for trying full parsing is that current information retrieval and IE methods are not scalable enough. Today, extraction of a fact is done by syntactic (surface form) pattern matching against all possible ways of expressing that fact. That means that for every type of fact (relation) many handmade patterns are needed, and this technique is too expensive when the number of different relations gets bigger.

The Yakushiji et al. system is based on a general purpose (domain independent) parser. The parser transforms each sentence into an argument structure (AS). Each AS contains a verb as the title, the semantic subject and

object(s) of the verb, and possibly adjective modifiers. The AS is a canonical structure, and that means that the parser has already taken care of all the variations that can occur in the text because of for example passivization and nominalization in the verbal phrases. The AS is comparable to the TUC Query Language (TQL) that is used in GeneTUC.

Next comes the domain specific part of the system. For each type of AS, a transformation rule (pattern matching) must be written, that converts the AS into a corresponding frame representation (FR). The FR is a possible end result of IE, and contains the semantics of the original verbal phrase. This technique scales better with large number of different relations, since the parser deals with the different syntactic ways of writing a verbal phrase, and only a few IE transformation rules must be written for each type of relation.

The article deals with three well-known problems of full parsing: Inefficiency, ambiguity and low coverage. These problems are partially solved with the use of pre- and post-processors. One pre-processor is the shallow parser. It introduces local constraints (a little stronger than Part-Of-Speech tagging) whenever possible in the text, and this increases the efficiency of the parser since obviously illegal (and computationally expensive) parse attempts can then be avoided. The other pre-processor is a term recognizer. It is not yet implemented, but it was simulated by hand-annotating the complex names in the sentences as units belonging to a given class. This gave a 10-fold increase in parsing speed, and also reduced the coverage problem since failure to recognize a complex term is often the reason that the parse fails.

The results of the experiment are not extremely good (23% success rate), but they give hope that this method can work (67% success rate) when more pre-/post-processing techniques are applied. 23% of the facts were uniquely (correctly) extracted. 24% of the facts were extracted with more than 1 possible FR (ambiguity) and 20% of the facts were extractable (without modifiers) from the partial results of the failed parses.

2.3 Goals

The goal of Information Extraction (IE) in the medical domain is as follows: We need to automate the task of IE from biomedical papers, because there are simply too many new papers every day for the researchers to keep up with. On the way to solving this goal many sub-problems must first be solved. Most of these sub-problems have already been identified by others, and are slowly being "solved" in terms of steadily increasing coverage and precision for the different methods.

Among the sub-problems that are slowly being solved are tagging (see the Brill tagger [2]) and "Term recognition". Term Recognition is important because clustering of long names/Noun Phrases in advance gives great improvement in speed and coverage/robustness of the full-parsers (see [15]). Specific systems are made by Bennett et al. [1], Cohen et al. [3], Fukuda et al. [7] and Proux et al. [12]. These systems report test results around 99% recall and 95% precision. They have already identified some major sources of precision problems and they also have specific plans to get rid of these problems as future work.

Before the mentioned sub problems are solved, the task of relation extraction has no hope of getting good precision or recall scores, but some researcher (like us) still try, believing that the basics will be solved soon enough. Current scores are comparable to what Pustejovsky et al. [13] reported at the Pacific Symposium on Biocomputing 2002 (PSB02) [24]. Their parser is a robust, shallow, corpus-based parser. Relational parsing means that they extract information on the form X relates to Y. In this case the specific relations are all inhibiting relations, and the X and Y can be entities (genes and proteins) or processes (e.g. binding). Their results are much better than previously published results, with 90% precision and 57% recall plus 22% partial recall. Partial recall means that just X or Y, but not both, was extracted. The way they get these good results is by their use of cascades of Finite States Automata (FSA), more or less in the same way that is done with local grammars [9] in Unitex [26]. One important step in getting good results was to realize that nominal-based relations (Predicative Nouns) had to be dealt with separately from normal verbal-based relations. All this work is a part of the Medstract project [23], building on the old Acromed system.

Another interesting approach to doing relation extraction is presented by Park [11]. They use a parser with combinatory categorial grammar to parse the relatively complex biomedical sentences, and they combine this with the corpus-based approach. In the end they do a gold standard test, with 48% recall and 80% precision, and these numbers are better than any other previously published comparable attempts. The conclusion of the article more or less agrees with Yakushiji et al. [15], in that full-parsing can be made to work, and it is worth the effort, because then we can extract more specific and meaningful facts from the abstracts. One example where full-parser usually performs better is anaphoric resolution, meaning the ability to recognize what is pointed to by terms such as "it".

2.4 Visualization

Visualization is another important area, because the biologists (end users) need to understand the information that is extracted from the biological texts. In the article "A literature network of human genes for high-throughput analysis of gene expression" [10], Jenssen et al. introduce a program called PubGene. It creates and visualizes an overview network of possibly related genes. The network

is built on the assumption that gene names co-occurring in Medline abstracts also have a related function or another relevant connection. The network is especially useful in Microarray experiments, because then many genes must be explored simultaneously. The methodology includes a database of gene names, a gene-to-article index, a gene-to-gene network, a gene network browser, and a gene expression and literature score. To handle the gene name problems the authors collected gene name variations from LocusLink [22], Human Gene Nomenclature Committee [21], the Genome Database [20] and GENATLAS [19]. The resulting gene identifier database contained 13712 different genes, and each became a node in the gene-to-gene network. Using the accumulated identifiers, the authors searched Medline and found 7512 co-occurring genes. Each co-occurrence linked two network nodes or added one to the weight of an existing link. The finished network allowed searches for individual nodes, resulting in a sub-network of the gene's closest neighbours, or an expression set from e.g. a Microarray experiment. The sub-networks of the searches indicate functional relations that the biologist should consider in her further work. Jenssen et al. proved their concept with a subset of well-known expressions. According to error analysis, most false positive errors stem from gene identifier problems, e.g. the gene names are too general.

The visualization of gene-interaction networks, e.g. as in [10], is very important for the biologists who are trying to understand what the role of a single gene is. Another field where visualization is very important is in the construction of local grammars (as in Unitex). The idea behind local grammars is that you cannot write general rules about how nouns and verbs combine into phrases and sentences, because there are simply too many irregularities or exceptions. In the end, you really need an exhaustive list of specific rules for every single possible use of a given verb: Normally accepted complements (e.g. nouns), all legal adverbial phrases for the verb, idiomatic uses with their allowed complement structures, and so on. This is an enormous work, since there is more than 10^{50} ways to build a sentence with at most twenty words [9], and therefore it is very important with a good visualization tool so that all these rules can be built fast with a minimum of extra work. The kind of local grammars described here are implemented in the visualization system Unitex [26].

Other examples of systems that include some sort of visualization are described in [4], [8], [11], and [15]. These articles describe complete approaches, with all the necessary steps from plain texts via knowledge bases to actually useful systems for the end users. They are written in the early stages of IE from biomedical papers, and they are giving general pointers and plans about what has to be done. It is also interesting that Internet is pointed out as a new kind of "Corpus" for IE systems to take advantage of, especially as databases such as Medline [25] become more accessible and structured.

3. Unitex

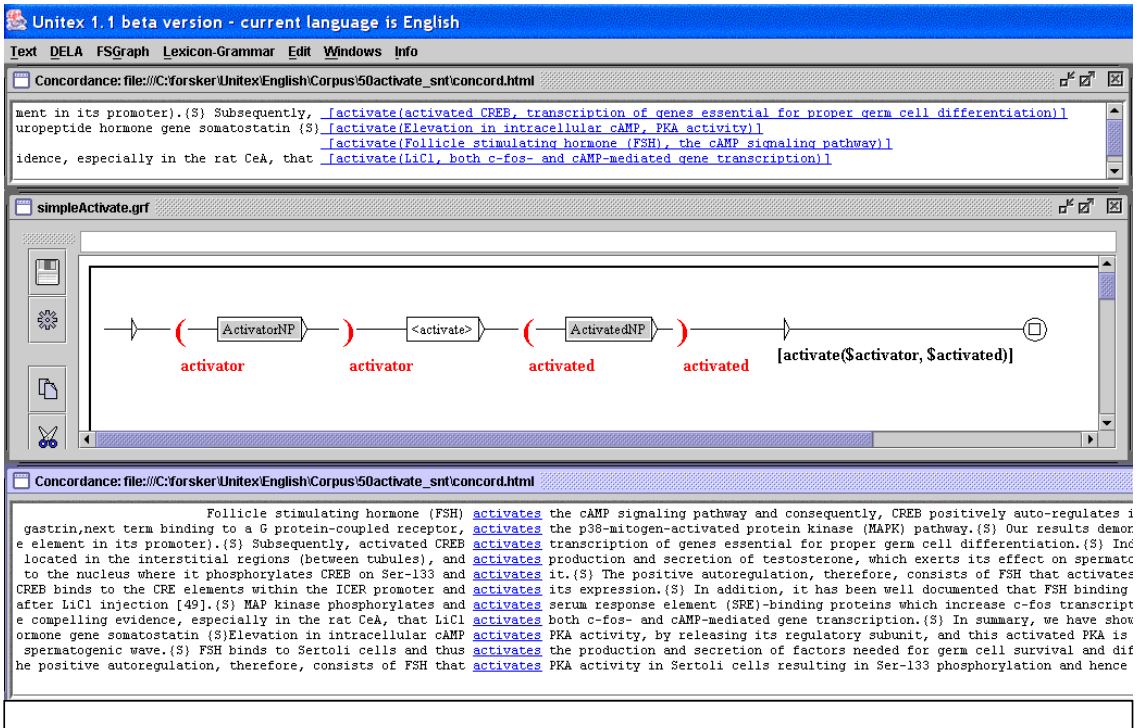

Figure 1, variables in Unitex

3.1 Variables

In Unitex graphs, variables can be used to produce just the output that is wanted. In Figure 1 two variables (*$activator* and *$activated*) is being used to transform the extracted facts from natural language into a very general predicate logic form.

The middle window in Figure 1 shows a graph were the sub-graph "GeneNameNP" is called two times. The text that matches the sub-graph the first time is stored in the *$activator* variable, and the "*Gene Noun-Phrase*" that matches the sub-graph the second time is store in the variable *$activated*. At the end is just an empty box that produces the desired output in the form of a logic predicate (*activate*) with two arguments, and the entire output is placed in angle brackets, in order to make it easier to separate it from the remaining text later.

The top window in Figure 1 shows a concordance structure with the results of applying this simple graph in *replace* mode, and the bottom window show a concordance structure for all occurrences of the word "activates" found in the 59 sentences.

3.2 Creating a context for "unknown words"

A simple way to speed up the process of classifying unknown words is by making a graph that contains one box with all the unknown words in it (See Figure 2, left side). When this graph is stored, it can later be applied for locating patterns in the text, and for building concordance structures, as it was done in Figure 2, bottom right side. This is very useful, since the unknown words are then highlighted and display together with their *context* (approximately the 12 nearest words) in the text. This can save a lot of time since unknown words are often declared (explicit or implicit) in the text where they are first used, and by using these definitions one can save valuable time that would otherwise have to be spent searching the Internet or other dictionary sources.

When searching for the 52 unknown words in the text, it was discovered that they constitute almost 4% of the total text. That means that every 26th (running) word is an unknown word.

"Stop words" are another challenge when using Unitex. Some very common words, like "a", is in some cases used as for example a name. If something is accidentally *named*

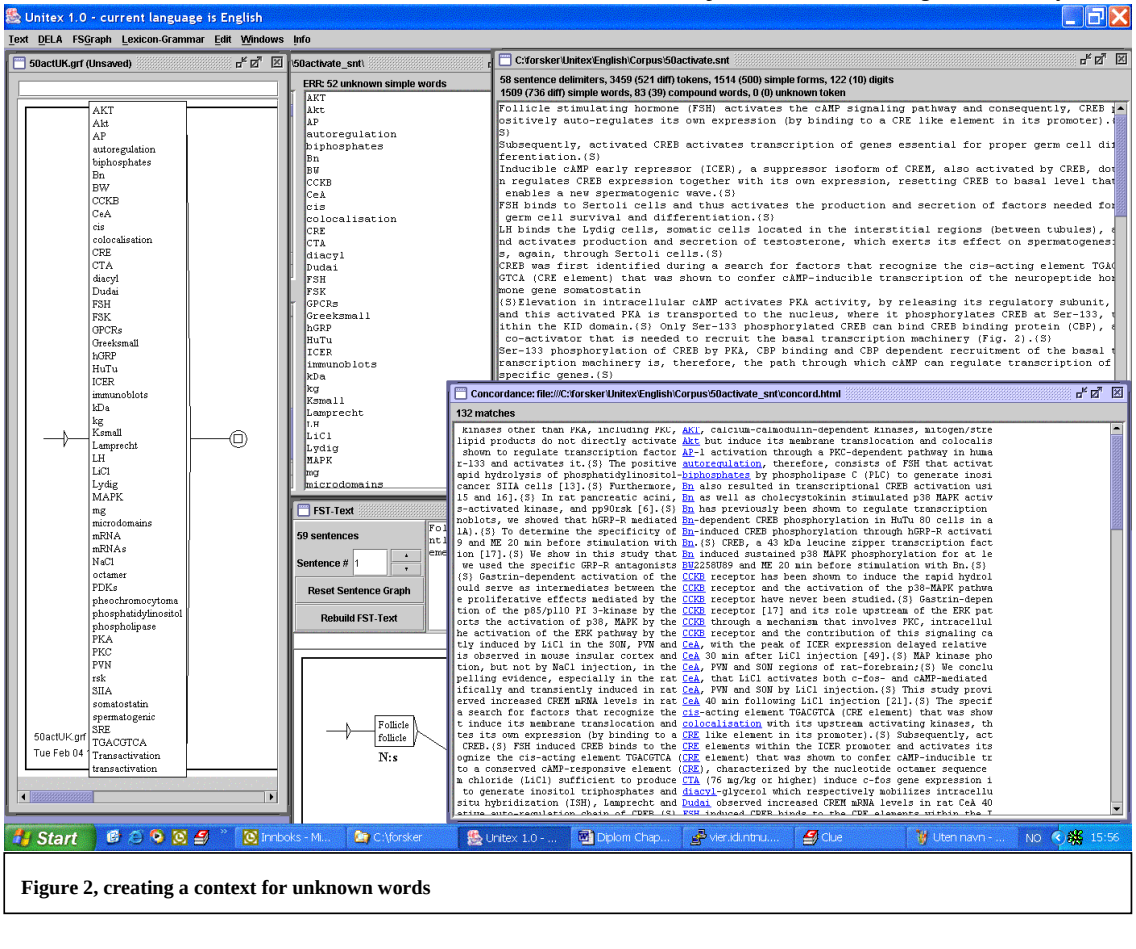

Figure 2, creating a context for unknown words

"a" it will cause a lot of confusion since a is *normally* used as a determiner in front of nouns, and not as a name. Luckily, there is a function in Unitex called Filter Dictionaries. They can be applied to the text before the standard dictionaries, and allows the removal of such "legal, but unwanted" interpretation of the words. All that is needed is a list of the words, and such lists exist already.

4. GeneTUC

In order to appreciate the results made with Unitex and local grammars, it is important to think about how they can be used later. In our case, Unitex's local grammars could be used to do effective pre-processing on the biomedical texts, before they are parsed by GeneTUC. This is a good idea because the strengths of Unitex (Fast processing, and a graphical interface) matches the weaknesses of GeneTUC (slow parsing, and no graphical interface). The reason for not doing *all* the work in Unitex is that GeneTUC already implements the entire framework for a tell-and-ask system, and the grammar is much more expressive than the regular automata in Unitex.

The end goal of GeneTUC is to build a question answering system that can be used by biologists to search out important bimolecular facts. Biomedical text, more often than not, contains Greek letters, punctuation in the middle of words, chemical formulae and so on. This constitutes a problem for GeneTUC, because it was made to deal only with standard ASCII-letters. The challenges of recognizing terms with strange characters and mapping them to their unique identifiers can be solved for example using Unitex. And once the compound nouns have been group intelligently in this manner, GeneTUC has a really good chance of parsing the entire sentence, just using its general domain-independent grammar.

After a sentence is parsed by GeneTUC, it is possible for the biologists to formulate questions about the sentences. Such questions will quite often be assigned a parse that is similar to the sentence where the answer to the question is given. In these cases the question form is stated in the grammar just as a transformation of a statement form, including movement of the subject/object categories, and addition of a *wh*-word. The question will also usually be more general or abstract than the statement containing the answer. For example, one might ask what *substance* activates a specific *gene*, and then finding out that the answer is a specific *protein*, which is indeed a subclass of the more abstract term *substance*.

In the cases where the question has a completely different form than the sentence containing the answer, certain rules are needed for GeneTUC to make the inferences that most humans do so easily. These rules can be formulated in a simplified form of plain English called Natural Readable Logic (NRL). This means that the biologists can write down exactly what they are thinking, e.g. when they infer that "phosphorylation of X" means "activation of X" (Figure 3). When enough such rules are made, GeneTUC will be a valuable tool for the scientists, since they can then get help finding the answers to important questions, without having to know most of the answer already, like you have to in current search engines.

5. Methods and Material

The current end goal for a working biomedical parsing system, whether it is GeneTUC, Unitex-based, or using any other parser, is to do automatic information extraction from the medical abstracts or full texts. In this project we are particularly interested in gene *activation*, and we want to extract information such as what gene/protein is the activator, what gene/protein is being activated, how reliable are the extracted facts, and what extra conditions must be satisfied. In order to start somewhere, a micro-biologist was asked to find around 50 sentences that contain facts about activation. Most of these sentences also contain the actual word "activate" in some form, but there are also a few sentences that use other words (e.g. X *confer transcription* of Y).

After local grammars were built for the given "activate-sentences", a test was run on a biological reference corpus, to see how general and applicable the graphs are.

Ser-133 phosphorylation of CREB by PKA

Figure 3, X Activation of Y by Z

5.1 Text Sources

This subsection describes the different sources that were used to acquire the text to parse. Most of the text is from the Medline abstracts database [25]. The first source contains an entire abstract that was previously used to train GeneTUC. The second source contains "random" single sentences selected by a biologist with the criterion that they should all contain facts about activation of a gene, protein or hormone. The third source is a 19.000 sentences large biological reference corpus that will be used for testing the finished local grammars in the end. This is the same text that has been used to test GeneTUC previously.

A Medline abstract about gastrin and CCK was used for preliminary testing, to see what the number of unknown words would be, and how ambiguous the sentences can be. The same abstract was also used to test GeneTUC in 2002, so it is possible to compare the results, and the amount of work that is needed to successfully "understand" a text using either of the two different systems.

Astrid Lægreid found more than 50 sentences describing the activation of different genes. Actually, the sentences are not only about *gene* activation. They also contain facts about *protein* and *hormone* activations. For the sake of

testing the Unitex methodology, it is not so important whether they are genes, proteins or hormones. Genes and proteins quite often have the same names anyway, since a protein is usually made from one or more corresponding genes. Hormones actually have slightly different names, but the sentences about hormone activation have the same form and context as the gene/protein activation sentences. That means that we can merge the gene, protein and hormone dictionaries, and just treat all these names as subjects or objects of the *activation* relation.

For the final tests we used the micro-biological reference corpus that was also used to test GeneTUC earlier (*abs2.txt*). It contains around 18.000 sentences, which is a little more than 5MB when it is stored in the Unicode format. Later a reference corpus with about 25.000 *tagged* tokens was acquired from "Centrum for Informations und Sprachverarbeitung" (CIS) at LMU in Munich. This reference corpus was also originally extracted from Medline, and it was used as a cross reference and aid during the classification of different medical words (primarily names) in this project. A program could be built that automatically classifies or suggest classification of words based on what tags they are given in such an *already tagged* reference corpus. This would save a lot of work, since every entity name would then only have to be manually processed one time. Right now, every researcher always has to start from scratch, and often ends up solving problems that have already been solved by others.

After all the graphs for the 59 activate sentences were finished, a test was run on "*abs2.txt*" to see how applicable the graphs were.

6. Results and Discussion

The Unitex graphs that were built had to be general enough to also accept sentences that are not *explicitly* programmed. That means that if we have training examples such as "X activates A", "X activates B" and "X activates D", then the very similar sentence "X activates C" should also be recognized by the system. This means that we have to introduce abstract graphs such as "X activates <Noun>", but if too many such abstractions are introduced, the system will end up also recognizing incorrect or "false" sentences. The results from the tests and the lessons learned during the graph building working phase will be summarized here.

6.1 Different Stages

The building of the graphs went through three more or less well defined iterations. The first iteration felt like putting different sentences together almost at random, but it was soon discovered that many sentences were too long for all the words to fit beside each other in one graph page, so the second iteration consisted of constructing sub graphs to cluster groups of words together and represent them as just one box (sub graph). These graphs were made so that words that often stood together in different sentences were put into the same sub graph. That allowed entire sentences to be represented in the main (top node) graph while still maintaining the desired left-to-right reading property. As the number of sub graphs grew, it became obvious that a good naming scheme was needed. It took some time to work this out, and that means that some graphs had to be completely rearranged later, and a few of the graph names had to be changed. It is a good idea to avoid this, because it will definitely introduce some new errors into the system, e.g. sentences that were recognized by the old graph might not be recognized by the new graph, and it will often take a lot of debugging to figure out exactly why. This phenomenon also happens for example when a function name is being changed in the code of a big program: When the name of a function is changed, all the places that call this function must also be updated. There is no support for such name changes in Unitex, so a lot of time will be spent doing this manually, and it is very easy to miss something, and get strange errors.
The third iteration was caused by the fact that too many different semantic meanings often ended up in a single graph, making it hard to extract meaningful facts from these specific graphs later. So another rearrangement consisting of splitting the graphs with ambiguous meanings into separate disambiguate graphs had to be done. This caused the "height" of the graphs (number "of lines" or "parallelism" in the graph) to increase, since different paths leading into one "ambiguous" box, now had to go to new separate disambiguated boxes. That also means that some boxes had to be duplicated, which is generally not a good thing, because then all subsequent work on the specific boxes must also be duplicated. Still, this is necessary, since the semantic output from the different boxes in the end must be different. For example "*activation of X*" can mean that X is being activated in one sentence, but that X is the activator in another sentence.

6.2 Naming Scheme

Different abstractions were tried in order to find good names for the sub graphs during the construction work. This consisted of splitting the sentences into well defined semantic and/or POS-based units. This turned out to be harder than expected, because quite often there would be an overlap between the units, and this would often be discovered long after the choice was made and the graph already constructed. The most successful abstractions were those including Gene Name Noun Phrases (POS), and the Activator/Activated (Semantic) sub-graphs. Since Protein/Gene name discovery in biomedical texts is considered a more or less solved problem [5], it is not necessary to be too careful about the explicit content of these graphs. In this work, these graphs were simply manually filled with the explicit coding of the names as they appeared in the text. For the sake of building a complete system later, it is very important to find one of

these systems that does protein/gene name discoveries in medical texts, because the current solutions are either too slow (manually coding every entry) or not accurate enough (importing *probable* entity names from nomenclature resources on internet).

Another very successful abstraction/naming scheme for the graph-building work was to make separate sub graphs for every *prepositional phrase* (PP), based on what leading preposition they contained. This was very practical when new sentences were added into the graph system, because one only had to identify the prepositions of the sentence, and then it was already obvious how the sentence should be split into sub graphs. The problem with this approach was that it sometimes led to "collisions" with the "*gene name, activator and activated*" naming scheme. Many sentences are on the form shown in Figure 3, and then a choice must be made whether "of Y by Z" should be coded by the "of" and "by" PP-graphs, or by the "activated" and "activator" sub-graphs. Regardless of what choice is being made, these different forms should be located close to each other in the parent graph, to ease the work later of debugging and add-ons to the system.

Another problem with the PP-graphs naming abstraction became evident later, as semantics were incorporated into the graphs. For example, one PP-graph called "*inPP*", contains prepositional phrases with very different semantic meanings, and the only thing they have in common is the fact that all this phrases start with the word "in". Because the semantics of these phrases are so different from each other, it would be better to spread them across different graphs. This is already partly done for example with the "InResponseToPP" graph.

References

[1] Bennett, H. A., He, Q., Powell, K., and Schatz, B. R. (1999) **Extracting noun phrases for all of Medline**. In *AMIA '99 (American Medical Informatics Assoc) Conf*, Washington, DC.

[2] Brill, E. (1992) **A simple rule-based part-of-speech tagger**. In *Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing*, Trento, Italy pages 152-155

[3] Cohen, K. B., Dolbey, A. E., Acquaah-Mensah, G. K. and Hunter, L. (2002) **Contrast and variability in gene names**. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain*. Pages 14-20

[4] Collier, N., Park, H., Ogata, N., Tateishi, Y., Nobata, C., Ohta, T., Sekimizu, T., Imai, H., Ibushi, K. and Tsujii, J. (1999) **The GENIA project: corpus-based knowledge acquisition and information extraction from genome research papers**, in *Proc. of the Annual Meeting of the European Association for Computational Linguistics (EACL-99)*, pp.271-272, Norway

[5] Eriksson, G., Franzén K., Olsson, F., Asker, L., Lidén, P. (2002). **Using Heuristics, Syntax and a Local Dynamic Dictionary for Protein Name Tagging**. Human Language Technology Conference 2002, San Diego, USA. URL: http://www.sics.se/~fredriko/papers/hlt02_abstract.pdf

[6] Ohta, Y., Yamamoto, Y., Okazaki, T., Uchiyama, I., and Takagi, T. (1997) **Automatic Construction of Knowledge Base from Biological Papers**. In *Proc. of the Fifth International Conference on Intelligent Systems for Molecular Biology (ISMB'97)*, pages 218-225.

[7] Fukuda, K., Tsunoda, T., Tamura, A., and Takagi, T. (1998) **toward information extraction: Identifying protein names from biological papers**. In *Proc. of the Pacific Symposium on Biocomputing '98 (PSB'98), Hawaii*. Human Genome Center, Institute of Medical Science, University of Tokyo. Email: ichiro@ims.u-tokyo.ac.jp

[8] Hishiki, T., Collier, N., Nobata, C., Ohta, T., Ogata, N., Sekimizu, T., Steiner, R., Park, H. S., and Tsujii, J. (1998) **Developing NLP tools for genome informatics: An information extraction perspective**. In *Proc. of Genome Informatics*, pages 81-90, Tokyo, Japan. Universal Academy Press, Inc., Tokyo, Japan.

[9] Gross, Maurice. (1997) **the Construction of Local Grammars**. In *Finite-State Language Processing*, E. Roche & Y. Schabès (eds.), Language, Speech, and Communication, Cambridge, Mass.: MIT Press, pages 329-354

[10] Jenssen, T. K., Lægreid, A., Komorowski, J., and Hovig, E. (2001) **a literature network of human genes for high-throughput analysis of gene expression**. In *Nature Genetics*, 28(1):21-28

[11] Park, Jong C. (2001) **Using Combinatory Categorial Grammar to Extract Biomedical Information**. In *IEEE Intelligent Systems*, 6. Korea Advanced Institute of Science and Technology. URL: http://computer.org/intelligent/ex2001/x6toc.htm, pages 62-67

[12] Proux, D., Rechenmann, F., Julliard, L., Pillet, V., and Jacq, B. (1998) **Detecting gene symbols and names in biological texts: A first step toward pertinent information extraction.** In Miyano, S. and Takagi, T., editors, *Ninth Workshop on Genome Informatics*, volume 9, pages 72-80, Tokyo, Japan.

[13] Pustejovsky, J., Castaño, J., Zhang, J., Kotecki, M. and Cochran, B. (2002) **Robust Relational Parsing over Biomedical Literature: Extracting Inhibit Relations.** In *Proceedings of the Pacific Symposium on Biocomputing*, pages 362-373. Hawaii, USA

[14] Sætre, Rune. (2002) **GeneTUC**. Technical report, Department of Computer and Information Science, Norwegian University of Science and Technology, 2002. URL: http://www.idi.ntnu.no/~satre/genetuc/GeneTUC.pdf

[15] Yakushiji, A., Tateisi, Y., Miyao, Y. and Tsujii, J. (2001) **Event Extraction from Biomedical Papers Using a Full Parser**. In *Proceedings of Pacific Symposium on Biocomputing 2001*, pages 408-419, Hawaii, USA

[16] CiteSeer. URL: http://citeseer.nj.nec.com/

[17] Gene Ontology. URL: http://www.geneontology.org/

[18] Google. URL: www.google.com

[19] GENATLAS. URL: www.dsi.univ-paris5.fr/genatlas/

[20] The Genome Database. URL: http://gdbwww.gdb.org/

[21] The Human Genome Organisation (HUGO) Gene Nomenclature Committee. URL: http://www.gene.ucl.ac.uk/nomenclature/

[22] LocusLink. URL: www.ncbi.nlm.nih.gov/LocusLink/

[23] Medstract Project. URL: www.medstract.org

[24] Pacific Symposium on Biocomputing. URL: http://psb.stanford.edu/psb02/

[25] PubMed Medline. URL: www.pubmed.gov

[26] Unitex. URL: http://www-igm.univ-mlv.fr/~unitex/

Paper II

ProtChew: Automatic Extraction of Protein Names from Biomedical Literature.
Amund Tveit, Rune Sætre, Astrid Lægreid and Tonje Strømmen Steigedal.
In *ICDE Proc. 21st International Conference on Data Engineering (ICDE'05).*
Tokyo, Japan, April 3-4, 2005. IEEE Electronic Proceedings, pages 1161–1161.
DOI Bookmark:
http://doi.ieeecomputersociety.org/10.1109/ICDE.2005.268

ProtChew: Automatic Extraction of Protein Names from Biomedical Literature

Amund Tveit and Rune Sætre
Department of Computer and Information Science
NTNU, Norway
{amund.tveit,rune.satre}@idi.ntnu.no

Astrid Lægreid and Tonje Strømmen Steigedal
Department of Cancer Research and Molecular Medicine
NTNU, Norway
{astrid.laegreid,tonje.strommen}@ntnu.no

Abstract

With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. Due to incomplete biomedical information databases, the extraction is not straightforward using dictionaries, and several approaches using contextual rules and machine learning have previously been proposed. Our work is inspired by the previous approaches, but is novel in the sense that it is fully automatic and does not rely on expert tagged corpora. The main ideas are 1) unigram tagging of corpora using known protein names for training examples for the protein name extraction classifier and 2) tight positive and negative examples by having protein-related words as negative examples and protein names/synonyms as positive examples. We present preliminary results on Medline abstracts about gastrin, further work will be on testing the approach on BioCreative benchmark data sets.

1. Introduction

With the increasing importance of accurate and up-to-date protein/gene information databases and ontologies for biomedical research, there is a need to extract protein information from biomedical research literature, e.g. those indexed in Medline [20].

Methodologically these approaches belong to the *information extraction field* [5], and in the biomedical domain they range from learning relationships between proteins/genes based on co-occurrences in Medline abstracts [9] to *manually* developed protein information extraction rules [21] and protein name classifiers trained on *manually* annotated training corpora [2].

Examples of protein names in a textual context

1. "duodenum, a **peptone** meal in the"
2. "subtilisin plus leucine **aminopeptidase** plus prolidase followed"
3. "predictable hydrolysis of **[3H]digoxin-12alpha** occurred in vitro"

1.1. Research Questions

Two of the main issues in information extraction in general are: 1) how to *automate* the generation of *annotated training data* needed to create extraction rules and classifiers and 2) how to select *appropriate negative examples* that are closely related but disjoined from the positive examples in order to ensure high accuracy for the information extraction of protein names. This leads to the following hypotheses:

1. Can existing *protein information databases* be used for *fully automatic generation of tagged training data* for protein name extraction classifiers?
2. Can existing *protein information databases* be used to create *appropriate negative examples* for information extraction of protein names?

The rest of this paper is organized as follows. Section 2 describes the materials used, section 3 presents our method, section 4 describes related work, section 5 presents empirical results, section 6 discusses our approach, and finally section 7 contains the conclusion and future work.

2. Materials

The materials used included biomedical (sample of Medline abstract) and general English (Brown) textual corpora, as well as protein databases, see below for a detailed overview.

As subject for the expert validation experiments we used the collection of 12.238 gastrin-related Medline abstracts that were available in September 2004. Gastrin was selected to fit the field of expertise of the researchers who evaluated our findings.

As a source for finding known protein names we use a web search system called Gsearch, developed at Department of Cancer Research and Molecular Medicine at NTNU. It integrates three common online protein databases, namely Swiss-Prot, LocusLink and UniGene.

The Brown repository (corpus) is an excellent resource for training a Part Of Speech (POS) tagger. It consists of 1,014,312 words of running text of edited English.

3. Our Approach

We have taken a modular approach where every submodule can easily be replaced by other similar modules in order to improve the general performance of the system. The main modules correlate with the main tasks that have to be solved in an information extraction setting. There are four modules connected to the data gathering phase, namely data selection, tokenization, POS-tagging and Stemming. Then three modules deal with classification, namely Gsearch, feature extraction and Classification. The last three modules are evaluation modules that handle cross-validation, expert evaluation and dataset statistics. See figure 2.

1. **Data Selection** The data selection module uses PubMed Entrez online system to return a set of PubMed IDs (PMIDs) and abstracts for a given protein, in our case "gastrin" (symbol GAS).
2. **Tokenization** The text is tokenized to split it into meaningful tokens, or "words". We use the WhiteSpaceTokenizer from NLTK. Words in parentheses were clustered together and tagged as a single token with the special tag *Paren*.

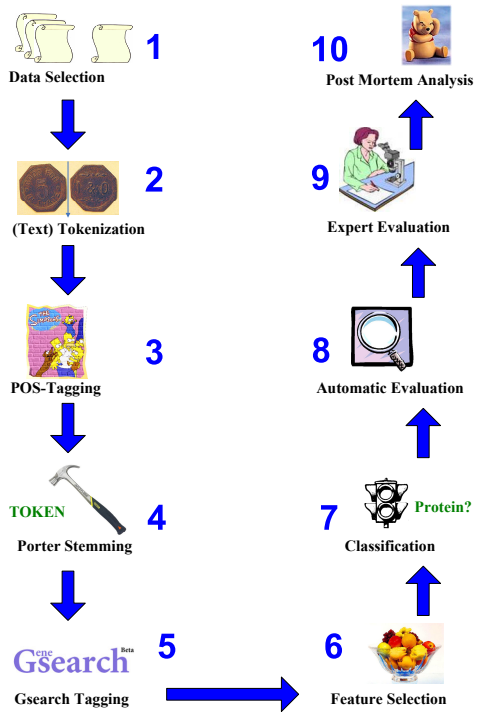

Figure 1. Overview of Our Approach

3. **POS tagging** using a Brill tagger trained on the Brown Corpus. This module acts as an advanced stop-word-list, excluding all the everyday common American words from our protein search. Later, the actual POS tags are also used as context features for their neighboring words.
4. **Porter-Stemming** If the stem of a word can be tagged by the Brill tagger, then the word itself is given the special tag "STEM", and thereby transferred to the common word list.
5. **Gsearch.org** tagging is our way of *automatically* creating positive and negative examples for the protein name extraction stage. Classifiers in general follow the rule "garbage in equals garbage out". One way to improve this is to do careful feature selection (out of the scope of this paper). Another is in the selection of *positive* and *negative* training data - which is what we are focusing on. The idea is that if an information extraction clas-

sifier should be able to discern between protein names and other entities, it in particular needs to handle entities that are as close to protein names as possible, i.e. *protein-related* entities. We select negative examples (i.e. protein-related entities) by using words (not filtered out by past modules) describing proteins, and positive examples by using protein names and synonyms. The proteins, synonyms and corresponding descriptions are found using the Gsearch.org search engine. It enables simultaneous searches in the Swiss-Prot, UniGene and LocusLink protein databases.

The remaining words are the untagged words that need to be classified (with the classifier trained on the positive and negative data generated in this step).

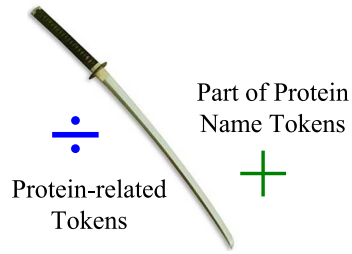

Figure 2. "Sharp edge" between positive and negative examples might improve classification accuracy

6. **Feature Selection** The features we use are the word itself (TEXT), the given tag (POS) from Brill or Gsearch (or None if the word is untagged), and other True/False features like HASBRACKET, HASFIRSTUPPER, HASNONALPHANUMPREFIX, ISLOWERCASE, ISNUMERIC, ISUPPERCASE. The features are collected for the word in question, and for the n nearest neighbors (we use $n = 3$ in our experiments).
7. **Classifier Performance** The positive and negative examples connected with the features described above are then used as training data for classification of *untagged* tokens as part of a protein name or not. Our selection of classifiers is quite pragmatic due to the *no free lunch theorem* [7], i.e. "there is no best classifier for all problems". We used the following classifiers: *Support Vector Machines* (with lin., pol., sig. and rbf kernels) in the SVM-Light tool [11], *Naive Bayes* in the Orange tool [6] and a *Proximal Support Vector Machine* (PSVM) in the Incridge tool [19] (PSVM is also known as Regularized Least Squares Classification)
8. **Automatic Evaluation** In order to efficiently test our extraction approach we first try to classify known data. If this gives extremely poor results there is no reason to pursue in classifying *untagged* tokens. The methods applied were "train and test" sets of 2500 examples each with various feature set combinations, as well as 10-fold cross-validation in order to test whether the "train and test"-set approach was ok.
9. **Expert Evaluation** The whole purpose of the extraction approach is to find proteins among *untagged* tokens. In order to do this we gave a sample of untagged tokens and their surrounding textual context to molecular biologists[1] so they could say if each token was a part of a protein name or not. We then used this as the golden standard to test our classifier performance and to measure true/false positives/negatives and to calculate F-Score and classification accuracy.
10. **Post Mortem Analysis** In order to characterize the size of the *untagged* protein names problem, we used the expert tagging from the molecular biologists in order to estimate a confidence interval for i) the probability of an untagged token being part of a protein name, and ii) the probability of a token being untagged, given our tagging sources.

4. Related Work

Our specific approach was on using existing databases to *automatically* annotate information extraction classifiers in biomedical corpora, and at the same time using these databases to create both positive and *negative examples*. We have not been able to find other work that does this, but there are quite a few approaches on extracting protein names from biomedical literature. Below, a brief overview is given. See [17] for a more comprehensive overview.

Bunescu et al. present a method *similar to ours*, except that they train their classifiers on *manually* created corpora [2, 3, 4]. Ginter et al. describe a method weighting words by positions for resolving gene/protein name disambiguation, but they use a *manually* developed corpus for training [8]. Bickel et al. describe an

1 co-authors

Acronym	Description
F1	3 neighbors w/all
F2	3 neighbors w/text
F3	3 neighbors w/text & POS
F4	3 neighbors w/POS & word-has-bracket

Table 1. Feature approaches

approach using Support Vector Machine classifiers for gene name recognition, but it is also trained using a *manually* generated biomedical corpus [1].

Mukherjea et al. describe a method that combines *manually* generated rules with rules learned using UMLS to do biomedical information extraction [12]. Torii and Vijay-Shanker use an unsupervised bootstrapping technique from Word Sense Disambiguation [18]. This resembles our approach in the sense that it is *fully automatic*, but differs in the sense that they use an unsupervised bootstrapping technique on names found using the *manually* developed rules presented in [13]. Jiampojamarn et al. describe a supervised method using comprehensive domain knowledge and dictionaries together with classifiers for biological term extraction [10].

5. Empirical results

Since our motivation is to test the feasibility of 1) automatic creation of training data for protein name classifiers and 2) selection of appropriate negative examples in the training data, we did not put much emphasis on the optimal selection of features for the information extraction classifiers. That is a natural next step, but outside the scope of this paper. The different feature sets we used are described in table 1, and more details about the features are given in *our approach*.

5.1. Automatic Evaluation

In order to get an overview of which classifier performance to expect, we first tested them on already tagged data, using protein names and symbols found in Gsearch as positive examples and other words from Gsearch (assumed to be protein-related) as negative examples (results in table 2). The data was first divided into a training and test set with 2500 examples each, and later we did a 10-fold cross-validation (XV) on all 5000 examples (train+test set) to verify the train and test approach.

Classifier	F1	F2	F3	F4
Majority	75.9	75.9	75.9	75.9
SVM Lin. t	75.9	75.9	75.9	75.9
SVM Pol.	**76.4**	75.9	75.9	75.9
SVM RBF	76.1	75.9	75.9	75.9
SVM Sig.	75.7	75.9	75.9	75.9
PSVM($\nu = 100$)	*68.0*	N/A	N/A	N/A
PSVM($\nu = 1$) XV	74.2	N/A	N/A	N/A

Table 2. Automatic Evaluation Results

Classifier	TP/TN	FP/FN	Prec/Rec/F	CA
N.Bayes	**6**/*120*	*67*/**7**	8/**46**/27	*63*
Majority	*0*/**187**	**0**/*13*	*NA/0/NA*	**94**
SVM Lin	*0*/**187**	**0**/*13*	*NA/0/NA*	**94**
SVM Pol	**6**/159	28/**7**	18/**46**/**32**	83
SVM rbf	3/174	13/10	**19**/23/21	89
SVM Sig	*0*/186	1/13	0/*0*/*NA*	93

Table 3. Protein classification - untagged words

5.2. Expert Evaluation

The main purpose of our extraction approach is to detect which untagged words that are part of protein names. In order to do (and test) this, we first tagged using the Brown Corpus (regular English words) and Gsearch (protein names and protein related words) and then we selected a sample of 200 words that *had not been tagged*. These words and their corresponding textual contexts were classified using the classifier, and compared to manual annotations done by biologists (table 3).

5.3. "Post Mortem" Analysis

In order to say something more general about the number of protein names that cannot be tagged with LocusLink, Swiss-Prot and UniGene, we used the results after stage 5 (Gsearch tagging) and the expert's classifications of untagged words. We created confidence intervals for the probability of a word being untagged after stage 5, and for the probability that an untagged word is a part of a protein name.

The total number of unique tokens in the 12000 abstracts covering *gastrin* is $N = 76359$, and 26885 of them were untagged. This gives an estimated probability of an untagged token $p_u = 26885/76359 = 35.21\%$ and $\sigma_u = \sqrt{(\frac{p_u(1-p_u)}{N})} \approx 0.0017$. The 95% confi-

dence interval is $[0.3521 - 1.96 \times 0.0017,\ 0.3521 + 1.96 \times 0.0017] \approx [34.88\%,\ 35.54\%]$

The expert found 13 protein names among a random sample of $n = 200$ untagged tokens (random sample from 26885 unique untagged tokens in total), this gives an estimated probability that an untagged word is a part of protein name $p_p = 13/200 = 6.5\%$ and $\sigma_p = \sqrt{(\frac{p_p(1-p_p)}{n})} \approx 0.0173$. The 95% confidence interval of $[6.5 - 1.96 \times 1.73,\ 6.5 + 1.96 \times 1.73] = [3.11\%,\ 9.89\%]$

6. Discussion

In the following section we discuss our approach on a step-by-step level (steps as presented in figure 2).

1. **Data selection** Since the results were inspected by cancer researchers the focus was naturally on proteins with a role in cancer development, and more specifically cancer in the stomach. One such protein is gastrin, and even though a search in the online PubMed Database returned more than eighteen thousand abstract IDs, only twelve thousand of these were found in our local academic copy of Medline. Another important question is if the gastrin collection is representative for Medline in general or for the "molecular biology" part of Medline in particular?

2. **Tokenization into "words"** The tokenization algorithm is important in the sense that it dictates which "words" you have to deal with later in the pipeline. How to deal with parentheses is another question. Sometimes they are important parts of a protein name (often part of the formula describing the protein), and other times they are just used to state that the words within them are not that important. We decided to keep the contents of parentheses as a single token, but this kind of parenthesis clustering is a hard problem, especially if the parentheses are not well balanced (e.g. smiley and "1), 2), 3)" style paragraph numbering). Parentheses in Medline are usually well balanced, though, so only very few tokens were missed because of erroneous clustering. Other tokens that require special attention are the multi-word-tokens. They can sometimes be composed using dash, bracket etc. as glue, and are at other times just normal single words separated with space, even though they should really be (grouped as) a single token. An example is protein names, such as "g-protein coupled receptor (GPCR)".

3. **a) Brown Corpus and tagging** We used the Brown Corpus, an American English corpus. It is rather old (1961) and maybe not completely representative of "Medline English". There is also the challenge of how quote symbol and apostrophes are used for protein names in Medline abstracts, e.g. as a marker for the five-prime or three-prime end of a DNA formula. Also, there are only one million words in the corpus, so not all lowercase and capital letter combinations of every word are present.

   b) POS tagging with Brill algorithm and the Brown Corpus The Brill tagger does not tag perfectly, so maybe classifier-based taggers such as SVM could perform better. The performance of the Brill tagger could be better if we used a higher-ordered tagger than the unigram tagger as input to Brill, but the memory need for n-gram taggers are $O(m^n)$, where m is the number of words in the dictionary. So with million word training- and test sets, even the use of just a bi-gram tagger gets quite expensive in terms of memory and time-use. Tagging itself may also introduce ambiguous tags (e.g. superman is a protein, but it may be tagged as a noun/name earlier in the pipeline, because that is the most common sense mentioned in the Brown Corpus).

4. **Porter-stemming** turns out to work poorly on protein and biological names, since they are often rooted in Latin or have acronyms as their name or symbol. E.g. the symbol for gastrin is GAS, and the porter stem of GAS becomes GA, which is wrong, and too ambiguous.

5. **Gsearch** The indexing algorithm of Gsearch also contains some stemming of search terms, leading to some "strange" results when creating the positive and negative training examples. The protein names found also cover "regular words" leading to other ambiguity problems, for example when "legal" protein names are removed earlier in the pipeline by the Brill tagger. Another weakness of Gsearch is that it is not "complete enough" (yet). It should be extended with a larger selection of databases and dictionaries covering biological terms, so that protein names like "pentagastrin" could also be found in the database.

6. **Feature Selection** Features in Information Extraction are usually ad-hoc and fosters creativity. It seems that all the ones we created made some sense, and that all the features took part in optimizing the classification of "untagged" test protein names.

7. **Classifier performance** Our selection and tuning of classifiers was quite limited due to our pri-

mary focus on *automatic generation of training data* as well as *ensuring high quality of the negative examples.* An opportunity for further improvement is to try other classifiers, e.g. C5.0, the Maximum Entropy classifier or the Instance-Based classifier.

8. **Automatic Evaluation** The automatic evaluation used two approaches: train+test set (SVM, Majority and PSVM) and 10-fold cross validation (PSVM). They gave ok, though not incredibly accurate, results. Natural improvements are to incorporate more (available) domain knowledge and additional features. An interesting observation is that the Majority and SVM classifiers always gave the *same accuracy* when syntactic clues (e.g. HASFIRSTUPPER) were left out. This is probably because our naive features do not catch the essence of protein names and their context.

9. **Expert evaluation of untagged data** Even though our (part- of-) protein name classification accuracy is relatively high ($\approx 80 - 90\%$), the results for most of the classifiers provide low precision and recall. The most promising classifier, from a precision and recall (F-Score) perspective, is Support Vector Machines with a Polynomial Kernel. It gives relatively many true positives, that can later be used as input for more advances natural language parsing techniques, like the ones used in [15]. Precision and recall results also suffer from being very unbalanced (i.e. relatively few protein names compared to the number of untagged words). In order to improve precision and recall results (as well as accuracy) we probably also need to improve filtering of untagged words *before* they become candidates for being part of protein names.

   We also know for a fact that our selection of positive and negative examples using protein information databases leads to slightly biased training data, since many common biological words are part of protein names (not found in our English corpus). This may lead to ambiguities when processing the corpus since most of the occurrences of these biological words are *not* part of protein names. In further work we could potentially gain from adding the bootstrapping methods for handling disambiguation presented in [18]. Testing our approach on benchmark datasets, like GENIA, would be a natural next step.

10. **Post mortem Dataset characterization** *Why finding protein names at all? are not they all in the major protein information databases?* No, we found that approximately between 1 and 3 % of *all* words found in our gastrin abstract selection were protein names (by multiplying the two confidence interval boundaries presented). If these estimates are representative for other biomedical texts in Medline this means that this problem is rather large.

Acknowledgements

We would like to thank Waclaw Kusnierczyk and Tore Amble for continuous support. And finally a thanks to the Gsearch developers Jo Kristian Bergum, Hallgeir Bergum and Frode Jünge.

7. Conclusion and Future Work

This paper presents a novel method for automatically creating both positive and negative training data for protein name extraction classifiers. Since we focused on the automatization of creating training data and relevant negative examples, we only used relatively simple domain modeling and feature extraction/selection approaches. This leads to promising, though not yet highly accurate, empirical results. So in the next round we need additional work on i) feature extraction and selection, and ii) incorporating domain knowledge. The approaches presented in [10, 12] seems to be complementary to ours and might increase accuracy in future versions of ProtChew.

To sum up the contributions:

1. **fully automatic extraction of protein names**
2. **"tight" negative examples using existing protein information databases (served by Gsearch.org) in order to get a "sharp classifier"**

Opportunities for future work are:

- improved tokenization (splitting on space and punctuation characters is not good enough.)
- stemming (the Porter algorithm for English language gives mediocre results on biological terms.)
- improved detection of sentence boundaries might be used to get more accurate context boundaries for classification of terms (splitting on punctuation characters gives slightly erroneous results). One possibly approach could be to train classifiers for sentence boundary detection on the Brown Corpus, or better on the GENIA biomedical corpus.
- combine the presented approach with traditional search engines such as Google as an additional information source about protein names e.g. as a feature for the input classifier, [16].

- part-of-speech tagging of Protein names (the complete protein names, frequently combined of many words) and then use inductive algorithms in order to find common grammars of protein names can potentially increase accuracy on detecting complete protein names, and not only part of protein names as we have focused on this work. (We have found that protein names often look like small sentences themselves, [14]).
- strongly improve the evaluation of our approach by applying it on the BioCreative datasets (http://www.mitre.org/public/biocreative/).

References

[1] S. Bickel, U. Brefeld, L. Faulstich, J. Hakenberg, U. Leser, C. Plake, and T. Scheffer. A Support Vector Machine classifier for gene name recognition. In *Proceedings of the EMBO Workshop: A Critical Assessment of Text Mining Methods in Molecular Biology*, March 2004.

[2] R. Bunescu, R. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani, and Y. W. Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.

[3] R. Bunescu, R. Ge, R. J. Kate, R. J. Mooney, Y. W. Wong, E. M. Marcotte, and A. K. Ramani. Learning to Extract Proteins and their Interactions from Medline Abstracts. In *Proceedings of the ICML-2003 Workshop on Machine Learning in Bioinformatics*, pages 46–53, August 2003.

[4] R. Bunescu, R. Ge, R. J. Mooney, E. Marcotte, and A. K. Ramani. Extracting Gene and Protein Names from Biomedical Abstracts. Unpublished Technical Note, Machine Learning Research Group, University of Texas at Austin, USA, March 2002.

[5] J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.

[6] J. Demsar and B. Zupan. Orange: From Experimental Machine Learning to Interactive Data Mining. White Paper, Faculty of Computer and Information Science, University of Ljubljana, 2004.

[7] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2nd edition, 2001.

[8] F. Ginter, J. Boberg, J. Jarvinen, and T. Salakoski. New Techniques for Disambiguation in Natural Language and Their Application to Biological Texts. *Journal of Machine Learning Research*, 5:605–621, June 2004.

[9] T.-K. Jenssen, A. Lægreid, J. Komorowski, and E. Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.

[10] S. Jiampojamarn. Biological term extraction using classification methods. Presentation at Dalhousie Natural Language Processing Meeting, June 2004.

[11] T. Joachims. *Advances in Kernel Methods: Support Vector Learning*, chapter 11 - Making Large-scale SVM Learning Practical. MIT Press, 1999.

[12] S. Mukherjea, L. V. Subramaniam, G. Chanda, S. Sankararaman, R. Kothari, V. Batra, D. Bhardwaj, and B. Srivastava. Enhancing a biomedical information extraction system with dictionary mining and context disambiguation. *IBM Journal of Research and Development*, 48(5/6):693–701, September/November 2004.

[13] M. Narayanaswamy, K. Ravikumar, and K. Vijay-Shanker. A biological named entity recognizer. In *Proceedings of the Pacific Symposium on Biocomputing 2003*, pages 427–438, 2003.

[14] R. Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.

[15] R. Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU München, Germany, April 2003.

[16] R. Sætre, A. Tveit, T. S. Steigedal, and A. Lægreid. Semantic Annotation of Biomedical Literature using Google. In D. O. Gervasi, D. M. Gavrilova, D. Y. Mun, D. D. Taniar, D. K. Tan, and D. V. Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO 2005)*, volume 3482 (Part III) of *Lecture Notes in Computer Science (LNCS)*, pages 327–337, Singapore, May 9-12 2005. Springer-Verlag Heidelberg.

[17] H. Shatkay and R. Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.

[18] M. Torii and K. Vijay-Shanker. Using Unlabeled MEDLINE Abstracts for Biological Named Entity Classification. In *Proceedings of the 13th Conference on Genome Informatics*, pages 567–568, 2002.

[19] A. Tveit, M. L. Hetland, and H. Engum. Incremental and Decremental Proximal Support Vector Classification using Decay Coefficients. In *Proceedings of the 5th International Conference on Data Warehousing and Knowledge Discovery (DAWAK'2003)*, volume 2737 of *Lecture Notes in Computer Science (LNCS)*, pages 422–429. Springer-Verlag, 2003.

[20] L. Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.

[21] H. Yu, V. Hatzivassiloglou, C. Friedman, A. Rzhetsky, and W. J. Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Paper III

Semantic Annotation of Biomedical Literature Using Google.
Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal and Astrid Lægreid.
In *Proc. Data Mining and Bioinformatics (DMBIO) 2005.*
International Workshop, Singapore, May 9-12, 2005.
Lecture Notes in Computer Science (LNCS) 2005.
Volume 3482, Part III, pages 327–337.
Springer-Verlag GmbH.
ISSN: 0302-9743.
ISBN: 3-540-25862-0.

Semantic Annotation of Biomedical Literature Using Google

Rune Sætre[1], Amund Tveit[1,3], Tonje S. Steigedal[2], and Astrid Lægreid[2]

[1] Department of Computer and Information Science
[2] Department of Cancer Research and Molecular Medicine
[3] Norwegian Center for Patient Record Research, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway
{rune.saetre, amund.tveit}@idi.ntnu.no
{tonje.strommen, astrid.laegreid}@medisin.ntnu.no

Abstract. With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. Due to incomplete biomedical information databases, the extraction is not straightforward using dictionaries, and several approaches using contextual rules and machine learning have previously been proposed. Our work is inspired by the previous approaches, but is novel in the sense that it is using Google for semantic annotation of the biomedical words. The semantic annotation accuracy obtained - 52% on words not found in the Brown Corpus, Swiss-Prot or LocusLink (accessed using Gsearch.org) - is justifying further work in this direction.

Keywords: Biomedical Literature Data Mining, Semantic Annotation.

1 Introduction

With the increasing importance of accurate and up-to-date databases for biomedical research, there is a need to extract information from biomedical research literature, e.g. those indexed in MEDLINE [34, 33, 15]. Examples of information databases are LocusLink, UniGene and Swiss-Prot [24, 23, 3].

Due to the rapidly growing amounts of biomedical literature, the information extraction process needs to be (mainly) automated. So far, the extraction approaches have provided promising results, but they are not sufficiently accurate and scalable.

Methodologically all the suggested approaches belong to the *information extraction field* [8], and in the biomedical domain they range from simple automatic methods to more sophisticated, but manual, methods. Good examples are: Learning relationships between proteins/genes based on co-occurrences in MEDLINE abstracts (e.g. [16]), *manually* developed information extraction rules (e.g. [35]), information extraction (e.g. protein names) classifiers trained on *manually* annotated training corpora (e.g. [4]), and our previous work on classifiers trained on *automatically* annotated training corpora [32]).

O. Gervasi et al. (Eds.): ICCSA 2005, LNCS 3482, pp. 327–337, 2005.

Examples of biological name entities in a textual context

1. "duodenum, a **peptone** meal in the"
2. "subtilisin plus leucine **amino-peptidase** plus prolidase followed"
3. "predictable hydrolysis of **[3H]digoxin-12alpha** occured in vitro"

Semantic Annotation

An important part of information extraction is to *know* what the information is, e.g. knowing that the term "gastrin" is a protein or that "Tylenol" is a medication. Obtaining and adding this knowledge to given terms and phrases is called *semantic tagging* or *semantic annotation*.

1.1 Research Hypothesis

Our hypothesis is based on ideas from our preliminary experiments using *Google* to generate features for protein name extraction classifiers in [**?**], i.e. using the number of search hits for a word as a feature.

Fig. 1. Google is among the biggest known "information haystacks"

– Google is probably the world's largest available source of heterogeneous electronically represented information. *Can it be used for semantic tagging of textual entities in biomedical literature?* And if so, how?

The rest of this paper is organized as follows. Section 2 describes the materials used, section 3 presents our method, section 4 presents empirical results, section 5 describes related work, section 6 discusses our approach, and finally the conclusion and future work.

2 Materials

The materials used included biomedical (sample of MEDLINE abstract) and general English (Brown) textual corpora, as well as protein databases. See below for a detailed overview.

MEDLINE Abstracts - Gastrin-Selection

The US National Institutes of Health (NIH) grants a free academic licence for PubMed/MEDLINE. It includes a local copy of 6.7 million abstracts, out of the 12.6 million entries that are available on their web interface. As subject for the expert validation experiments we used the collection of 12.238 gastrin-related MEDLINE abstracts that were available in September 2004.

Biomedical Information Databases

As a source for finding already known protein names we used a web search system called Gsearch, developed at Department of Cancer Research and Molecular Medicine at NTNU. It integrates common online protein databases, e.g. Swiss-Prot, LocusLink and UniGene, [24, 23, 3].

The Brown Corpus

The Brown repository (corpus) is an excellent resource for training a Part Of Speech (POS) tagger. It consists of 1,014,312 words of running text of edited English prose printed in the United States during the calendar year 1961. All the tokens are manually tagged using an extended Brown Corpus Tagset, containing 135 tags (Lancester-OsloBergen-tagset). The Brown corpus is included in the Python NLTK data-package, found at Sourceforge.

3 Our Approach

We have taken a modular approach where every submodule can easily be replaced by other similar modules in order to improve the general performance of the system. There are five modules connected to the data gathering phase, namely data selection, tokenization, POS-tagging, Stemming and Gsearch. Then the sixth and last module does a Google search for each extracted term. See figure 2.

1. **Data Selection.** The data selection module uses PubMed Entrez online system to return a set of PubMed IDs (PMIDs) for a given protein, in our case "gastrin" (symbol GAS). The PMIDs are matched against our local copy of MEDLINE, to extract the specific abstracts.
2. **Tokenization.** The text is tokenized to split it into meaningful tokens, or "words". We use the WhiteSpaceTokenizer from NLTK with some extra processing to adapt to the Brown Corpus, where every special character (like ( ) " ' - , and .) is treated as a seperate token. Words in parentheses are clustered together and tagged as a single token with the special tag *Paren*.
3. **POS tagging.** Next, the text is tagged with Part-of-Speech (POS) tags using a Brill tagger trained on the Brown Corpus. This module acts as an advanced stop-word-list, excluding all the everyday common American English words from our protein search. Later, the actually given POS tags are used also as context features for the neighboring words.

Fig. 2. Overview of Our Approach (named Alchymoogle)

4. **Porter-Stemming.** We use the Porter Stemming Algorithm (also from NLTK) to remove even more everyday words from the "possibly biological term" candidate list. If the stem of a word can be tagged by the Brill tagger, then the word itself is given the special tag "STEM", and thereby transferred to the common word list.
5. **Gsearch.** Identifies and removes already known entities from the search, but after the lookup in Gsearch, there are still some unknown words that are not yet stored in our dictionaries or databases, so in order to do any reasoning about these words it is important to know which class they belong to. Therefore, in the next phase they are subjected to some advanced Google-searching, in order to determine this.
6. **Google Class Selection.** We have a network of 275 nouns, arranged in a semantic network on the form "X is a kind of Y". These nouns represent the classes that we want to annotate each word with. The input to this phase is a list of hitherto unknown words. From each Word a query on the form in the example below is formed (query syntax: *Word is (an|a)"*).

   Then these queries are fed to the PyGoogle module which allows 1000 queries to be run against the Google search engine every day with a personal

password key. In order to maximize the use of this quota, the results of every query are cached locally, so that each given query will be executed only once. If a solution to the classification problem is not present among the first 10 results returned, the resultset can be expanded by 10 at a time, at the cost of one of the thousand quota-queries every time.

Each returned hit from Google contains a "snippet" with the given query phrase and approximately 10 words on each side of it. We use some simple regular grammars to match the phrase and the words following it. If the next word is a noun it is returned. Otherwise, adjectives are skipped until a noun is encountered, or a "miss" is returned.

4 Empirical Results

The table below shows the calculated classification scores for the expert evaluation phase. The first column shows *correct* predictions (True Positives and Negatives), the second column shows *incorrect* predictions (False Positives and Negatives), the third column gives Precision and Recall, the fourth gives the standard (balanced) F-Score number, and the last column presents the overall classification accuracy (correct classifications vs. incorrect ones).

Table 1. Semantic classification of *untagged* words

Classifier	TP/TN	FP/FN	Prec/Rec	F-score	CA
Alchymoogle	24/80	31/65	43.6/27.0	33.3	52.0

5 Related Work

Our specific approach was on using Google for *direct* semantic annotation (searching for is-a relations) of tokens (words) in biomedical corpora. We haven't been able to find other work that does this, but Dingare et al. is on using the number of Google hits as input features for a maximum entropy classifier used to detect protein and gene names [10, 11]. Our work differs since we use Google to *directly determine* the semantic class of a word (searching for is-a relationships and parsing text (filtering adjectives) after *(a/an)* in *"Word is (a|an)*, as opposed to Dingare et al.'s *indirect* use of Google search as a feature for the information extraction classifier. A second difference between the approaches is that we search for explicit semantic annotation (e.g. "word is a protein") as opposed to their search for hints (e.g. "word protein"). The third important difference is that our approach does *automatic* annotation of corpuses, whereas they require pre-tagged (manually created) corpuses in their approach.

Other related works include extracting protein names from biomedical literature and some on semantic tagging using the web. Under, a brief overview of related work is given.

Work describing approaches for semantic annotation using the Web can be found in [27, 12, 18, 19, 9, 22].

Semantic Annotation of Biomedical Literature

Other approaches for (semantic) annotation (mainly for protein and gene names) of biomedical literature include:

- Rule-based discovery of names (e.g. of proteins and genes), [13, 29, 36, 35]
- Methods for discovering relationships of proteins and genes, [2, 16].
- Classifier approaches (machine learning) with textual context as features, [4, 5, 6, 14, 1, 20, 30, 21, 17]
- Other approaches include generating probabilistic rules for detecting variants of biomedical terms, [31]

A comprehensive overview of such methods is provided in [28].

The paper by Cimiano and Staab [7] shows that a system (PANKOW) similar to ours works, and can be taken as a proof that automatic extraction using Google is a useful approach. Our systems differ in that we have 275 different semantic tags, while they only use 59 concepts in their ontology. They also have a table explaining how the number of concepts in a system influences the recall and precision in several other semantic annotation systems.

6 Discussion

In the following section we discuss our approach step-by-step. (The steps as presented in fig. 2.)

1. **Data selection.** Since the results were inspected by cancer researchers the focus was naturally on proteins with a role in cancer development, and more specifically cancer in the stomach. One such protein is gastrin, and even though a search in the online PubMed Database returned more than eighteen thousand abstract IDs, only twelve thousand of these were found in our local academic copy of MEDLINE. Therefore only 12.238 abstracts were used as input to the tokenizer. Another important question is if the gastrin collection is representative for MEDLINE in general or for the "molecular biology" part of MEDLINE in particular.
2. **Tokenization into "words".** The tokenization algorithm is important in the sense that it dictates which "words" you have to deal with later in the pipeline. Our choice of using the Brown Corpus for training the Unigram and Brill taggers also influences our choice of tokenizing algorithm. For example, in the Brown Corpus all punctuation characters like comma, full stop, hyphen and so on are written with whitespace both before and after them. This turns them into separate tokens, disconnected from each other and from the other tokens. How to deal with parentheses is another question. Sometimes they are important parts of a protein name (often part of "formulae" describing the protein), and other times they are just used to state that the words within

them aren't that important. We decided to keep the contents of parentheses as a single token, but this kind of parentheses clustering is a hard problem, especially if the parentheses aren't well balanced (like smiley and "1), 2), 3)" style paragraph numbering). Parentheses in MEDLINE are usually well balanced, but still some mistokenization was introduced at this point. Other tokens that require special attention are the multi-word-tokens. They can sometimes be composed using dash, bracket etc. as glue, but are at other times single words separated with whitespaces, even though they should really be one single token. One example is protein names, such as g-protein coupled receptor (GPCR).

3. **a) Brown Corpus and tagging.** To train the Unigram and Brill taggers, an already tagged text is needed as a training set. We used the Brown Corpus, an American English corpus made from texts from 1961. They are rather old, and might not be as representative of "MEDLINE English" as we want. There is also the challenge of how quote symbols and apostrophes are used for protein names in MEDLINE abstracts, e.g. as a marker for the five-prime or three-prime end of a DNA formula. Also, there are only one million words in the corpus, so not all lowercase and capital letter combinations of every word are present.
   b) POS tagging with Brill algorithm and the Brown Corpus. The Brill tagger doesn't tag perfectly, so maybe classifier-based taggers such as SVM could perform better. The performance of the Brill tagger could be better if we used a higher-ordered tagger than the unigram tagger as input to Brill, but the memory need for n-gram taggers are $O(m^n)$, where m is the number of words in the dictionary. So with million word training- and test sets, even the use of just a bi-gram tagger gets quite expensive in terms of memory and time-use. Tagging itself may also introduce ambiguous tags (e.g. superman is a protein, but it may be tagged as a noun/name earlier in the pipeline, because that's the most common sense mentioned in the Brown Corpus).
4. **Porter-stemming.** turns out to work poorly on protein and biological names, since they are often rooted in Latin or have acronyms as their name or symbol. E.g. the symbol for gastrin is GAS, and the porter stem of GAS becomes GA, which is wrong, and too ambiguous.
5. **Gsearch.** The indexing algorithm of Gsearch also contains some stemming of the search terms, leading to some "strange" results when removing well-known proteins from the unknown words list. It should be extended with a larger selection of databases and dictionaries covering biological terms, so that protein names like "peptone" could also be found in the database. In other words there are "precision and recall" issues also at this stage, but our program should be able to solve "half of this problem" automatically. The worst problem is actually how to handle names with "strange characters" like ([]) in them, since these characters are usually not taken into account during the index-building in systems like Gsearch (or Google).
6. **Google Search.** The precision of (positive) classification and the total classification accuracy is close to 50%, which is really good considering that no

context information has been used in the classification process. By using context information in the way that is done in [**?**] it should be possible to increase the classification accuracy further. We had a lower recall than expected (24/89 = 27.0%), mainly because a lot of our unknown words are parts of a multi-word-tokens, and can only be sensibly classified using the context which contains the rest of the multi-word-unit. Also, many of the words are not nouns, so they are not suitable class names in the first place, but still expert biologists often think of them in a concrete way. One example of this is "extracardiac", which were tagged as a place (outside the heart), even though nobody would actually write "extracardiac is a place outside the heart". (Except, I just did! And that really illustrates the problem of freedom, when dealing with Natural Language Understanding.)

We did another test using 1500 semantic classes, instead of the 275 strictly molecular biology related classes. Then we got more hits among the 200 words, so this may be a method to increase the coverage of our system. It is of course much harder to manually evaluate these results, and there is also the danger of lowering the precision this way.

7 Conclusion and Future Work

This paper presents a novel approach - Alchymoogle - using Google for semantic annotation of entities (words) in biomedical literature.

We got empirically promising results - 52% semantic annotation accuracy ((TP+TN)/N, TP=24,TN=80,N=200) in the answers provided by Alchymoogle compared to expert classification performed by a molecular biologist. This encourages further work possibly in combination with other approaches (e.g. rule- and classification based information extraction methods), in order to improve the overall accuracy (both with respect to precision and recall). Disambiguation is another issue that needs to be further investigated. Other opportunities for future work include:

- Improve tokenization. Just splitting on whitespace and punctuation characters is *not* good enough. In biomedical texts non-alphabetic characters such as brackets and dashes need to be handled better.
- Improve stemming. The Porter algorithm for English language gives mediocre results on biomedical terms (e.g. protein names).
- Do spell-checking before a query is sent to Google, e.g. allowing minor variations of words (using the Levenshtein Distance).
- Search for other semantic tags using Google, e.g. "is a kind of" and "resembles", as well as negations ("is not a").
- Investigate whether the Google ranking is correlated with the accuracy of the proposed semantic tag. Are highly ranked pages better sources than lower ranked ones?
- Test our approach on larger datasets, e.g. *all* available MEDLINE abstracts.
- Combine this approach with more advanced natural language parsing techniques in order to improve the accuracy, [25, 26].

– In order to find multiword tokens, one could extend the search query *(" X is (an|a) ")* to also include neighboring words of X, and then see how this affects the number of hits returned by Google. If there is no reduction in the number of hits, this means that the words are "always" printed together and are likely constituents in a multiword token. If you have only one actual hit to begin with, the certainty of the previous statement is of course very weak, but with increasing number of hits, the confidence is also growing.

Acknowledgements

We would like to thank Waclaw Kusnierczyk for proposing additional biomedical information databases for inclusion in future work, and Tore Amble for continuous support. We would also like to thank Martin Thorsen Ranang for proposing improvements for future work. And finally a thanks to the Gsearch developers Jo Kristian Bergum, Hallgeir Bergum and Frode Jünge.

References

1. Steffen Bickel, Ulf Brefeld, Lukas Faulstich, Jrg Hakenberg, Ulf Leser, Condrad Plake, , and Tobias Scheffer. A Support Vector Machine classifier for gene name recognition. In *Proceedings of the EMBO Workshop: A Critical Assessment of Text Mining Methods in Molecular Biology*, March 2004.
2. C. Blaschke, MA. Andrade, C. Ouzounis, and A. Valencia. Automatic Extraction of biological information from scientific text: Protein-protein interactions. In *Proceedings of International Conference on Intelligent Systems for Molecular Biology*, pages 60–67. AAAI, 1999.
3. B. Boeckmann, A. Bairoch, R. Apweiler, MC. Blatter, A. Estreicher, E. Gasteiger, MJ Martin, K Michoud, C. O'Donovan, I. Phan, S. Pilbout, and M. Schneider. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic Acids Research*, 31(1):365–370, January 2003.
4. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents (Forthcoming)*, 2004.
5. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Raymond J. Mooney, Yuk Wah Wong, Edward M. Marcotte, and Arun Kumar Ramani. Learning to Extract Proteins and their Interactions from Medline Abstracts. In *Proceedings of the ICML-2003 Workshop on Machine Learning in Bioinformatics*, pages 46–53, August 2003.
6. Razvan Bunescu, Ruifang Ge, Raymond J. Mooney, Edward Marcotte, and Arun Kumar Ramani. Extracting Gene and Protein Names from Biomedical Abstracts. Unpublished Technical Note, Machine Learning Research Group, University of Texas at Austin, USA, March 2002.
7. Philipp Cimiano and Steffen Staab. Learning by Googling. *SIGKDD Explorations Newsletter*, 6(2):24–34, December 2004.
8. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.

9. Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, pages 178–186. ACM, 2003.
10. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, and Beatrice Alex. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. In *Proceedings of the BioCreative Workshop*, March 2004.
11. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. Submitted to BMC Bioinformatics, 2004.
12. Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. Submitted to Artificial Intelligence, 2004.
13. K. Fukuda, A. Tamura, T. Tsunoda, and T. Takagi. Toward information extraction: identifying protein names from biological papers. In *Proceedings of Pacific Symposium on Biocomputing*, pages 707–718, 1998.
14. Filip Ginter, Jorma Boberg, Jouni Jarvinen, and Tapio Salakoski. New Techniques for Disambiguation in Natural Language and Their Application to Biological Texts. *Journal of Machine Learning Research*, 5:605–621, June 2004.
15. Jun ichi Tsuji and Limsoon Wong. Natural Language Processing and Information Extraction in Biology. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 372–373, 2001.
16. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
17. Sittichai Jiampojamarn. Biological term extraction using classification methods. Presentation at Dalhousie Natural Language Processing Meeting, June 2004.
18. Vinay Kakade and Madhura Sharangpani. Improving the Precision of Web Search for Medical Domain using Automatic Query Expansion. Online, 2004.
19. Udo Kruschwitz. Automatically Acquired Domain Knowledge for ad hoc Search: Evaluation Results. In *Proceedings of the 2003 Intl. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*. IEEE, 2003.
20. Sougata Mukherjea, L. Venkata Subramaniam, Gaurav Chanda, Sriram Sankararaman, Ravi Kothari, Vishal Batra, Deo Bhardwaj, and Biplav Srivastava. Enhancing a biomedical information extraction system with dictionary mining and context disambiguation. *IBM Journal of Research and Development*, 48(5/6):693–701, September/November 2004.
21. M. Narayanaswamy, KE Ravikumar, and K Vijay-Shanker. A biological named entity recognizer. In *Proceedings of the Pacific Symposium on Biocomputing 2003*, pages 427–438, 2003.
22. David Parry. A fuzzy ontology for medical document retrieval. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32*, pages 121–126. ACM Press, 2004.
23. JU. Pontius, L. Wagner, and GD. Schuler. *The NCBI Handbook*, chapter UniGene: a unified view of the transcriptome. National Center for Biotechnology Information, 2003.
24. KD Pruitt and DR Maglott. RefSeq and LocusLink: NCBI gene-centered resources. *Nucleic Acids Research*, 29(1):137–140, January 2001.

25. Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.
26. Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU Munchen, Germany, April 2003.
27. Urvi Shah, Tim Finin, and Anupam Joshi. Information Retrieval on the Semantic Web. In *Proceedings of CIKM 2002*, pages 461–468. ACM Press, 2002.
28. Hagit Shatkay and Ronen Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.
29. Lorraine Tanabe and W. John Wilbur. Tagging gene and protein names in biomedical text. *Bioinformatics*, 18(8):1124–1132, 2002.
30. Manabu Torii and K. Vijay-Shanker. Using Unlabeled MEDLINE Abstracts for Biological Named Entity Classification. In *Proceedings of the 13th Conference on Genome Informatics*, pages 567–568, 2002.
31. Yoshimasa Tsuruoka and Jun'ichi Tsuji. Probabilistic Term Variant Generator for Biomedical Terms. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 167–173. ACM, July/August 2003.
32. Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from . In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, Tokyo, Japan, April 2005. IEEE Press (Forthcoming).
33. Limsoon Wong. A Protein Interaction Extraction System. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 520–530, 2001.
34. Limsoon Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.
35. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.
36. Hong Yu, Vasileios Hatzivassiloglou, Andrey Rzhetsky, and W. John Wilbur. Automatically identifying gene/protein terms in MEDLINE abstracts. *Journal of Biomedical Informatics*, 35(5/6):322–330, October 2002.

Paper IV

gProt: Annotating Protein Interactions Using Google and Gene Ontology.
Rune Sætre, Amund Tveit, Martin Thorsen Ranang, Tonje S. Steigedal, Liv Thommesen, Kamilla Stunes and Astrid Lægreid.
In *Proc. Knowledge-Based Intelligent Information and Engineering Systems (KES) 2005*. International Conference, Melbourne, Australia, September 14-16, 2005.
Lecture Notes in Artificial Intelligence (LNAI) 2005.
Volume 3683, Part III, pages 1195–1203.
Springer-Verlag GmbH.
ISSN: 0302-9743.
ISBN: 3-540-28896-1.

gProt: Annotating Protein Interactions Using Google and Gene Ontology

Rune Sætre[1], Amund Tveit[1,3], Martin Thorsen Ranang[1], Tonje S. Steigedal[2], Liv Thommesen[2], Kamilla Stunes[2], and Astrid Lægreid[2]

[1] Department of Computer and Information Science,
Norwegian University of Science and Technology,
N-7491 Trondheim, Norway
{rune.saetre,amund.tveit,martin.ranang}@idi.ntnu.no
[2] Department of Cancer Research and Molecular Medicine,
Norwegian University of Science and Technology,
N-7491 Trondheim, Norway
{tonje.strommen,liv.thommesen,kamilla.stunes,astrid.laegreid}@ntnu.no
[3] Norwegian Centre for Patient Record Research
Norwegian University of Science and Technology,
N-7491 Trondheim, Norway

Abstract. With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. Due to incomplete biomedical information databases, the extraction cannot be done straightforward using dictionaries, so several approaches using contextual rules and machine learning have previously been proposed. Our work is inspired by the previous approaches, but is novel in the sense that it combines Google and Gene Ontology for annotating protein interactions. We got promising empirical results - 57.5% terms as valid GO annotations, and 16.9% protein names in the answers provided by our system gProt. The total error-rate was 25.6% consisting mainly of overly general answers and syntactic errors, but also including semantic errors, other biological entities (than proteins and GO-terms) and false information sources.

Keywords: Biomedical Literature Data Mining, Gene Ontology, Google API

1 Introduction

With the increasing importance of accurate and up-to-date databases about proteins and genes for research, there is a need for efficient ways of updating these databases by extracting information from biomedical research literature [8, 20, 21], e.g. those indexed in MEDLINE. Examples of information resources containing such information are LocusLink, UniGene and Swiss-Prot for protein info and the Gene Ontology for semantic labels.

Due to the large and rapidly growing amounts of biomedical literature, the extraction process needs to be more *automatic* than previously. Current extraction approaches have provided promising results, but they are not sufficiently

R. Khosla et al. (Eds.): KES 2005, LNAI 3683, pp. 1195–1203, 2005.

accurate and scalable. Methodologically all the suggested approaches belong to the *information extraction field* [3], and in the biomedical domain they range from simple automatic methods to more sophisticated, but slightly more manual, methods. Good examples are: Learning relationships between proteins/genes based on co-occurrences in MEDLINE abstracts (e.g. [9]), *manually* developed information extraction rules (e.g. [22]), information extraction (e.g. protein names) classifiers trained on *manually* annotated training corpora (e.g. [1]), and classifiers trained on *automatically* annotated training corpora [19]).

1.1 Research Hypothesis

Internet Search Engines such as Google, Yahoo and MSN Search are the world's largest readily available information sources, also in the biomedical domain. Based on promising results from recent work on using Google for semantic annotation of biomedical literature [16], we are encouraged to investigate if Google can be used to find protein interactions that match the Gene Ontology (GO). This leads to the hypothesis:

Can Internet Search engines such as Google be used to annotate protein interactions in the Gene Ontology framework?

The rest of this paper is organized as follows. Section 2 describes the materials used, section 3 presents our method, section 4 presents empirical results, section 5 describes related work, section 6 discusses our approach, and last the conclusion and future work.

2 Materials

See fig. 1 for an overview of the system. As input for our experiments we used the following:

- 10 proteins that are already well-known to our biology experts.
- 37 verb-templates suggested by Martin et. al (LexiQuest) [12].

Proteins

The following proteins were used as input to the system.

Proteins used
'EGF', 'TNF', 'CCK', 'gastrin', 'CCKAR', 'CCKBR', 'p53', 'ATF1', 'CREB', 'CREM'.

In addition, each protein is also described by several other names or synonyms in the literature. E.g. gastrin is also known as 'g14', 'g17', 'g34', 'GAS', 'gast', 'gastrin precursor', 'gastrin 14', etc. So our biologists compiled a list of roughly 10 synonyms for each protein, *giving us about 100 terms total to annotate.*

Interaction Verbs

We selected our interaction verb templates from table 1 in [12]. They had a list of 44 verbs, but we chose to use only 37 of these verbs. The reason for this is that we are focusing on simple statements like "gastrin activates ...", with the object of the verb following directly after the verb template. The following table shows the original list of verbs, with the removed ones in parenthesis.

Verb templates used
acetylates, activates, (antagonizes), associates with, (attenuates), (binding to), binds, blocks, (bonds), (complex), deactivates, decreases, degrades, dephosphorylates, dimerizes, dissociates from, downregulates, forms complex with, hydrolyses, inactivates, increases, induces, inhibits, interacts with, links, mediates, (oligomerizes), overexpresses, phosphorylates, potentiates, precipitates with, reacts with, recruits, (reduces), regulates, releases, represses, stimulates, transactivates, transduces, transforms, triggers, ubiquitinates, upregulates,

3 Our Approach

We have taken a modular approach where every submodule can easily be replaced by other similar modules in order to improve the general performance of the system. There are five modules in the system. The first one sets up the search queries, the second runs the queries against Google, the third one tokenizes the results, the fourth parses the tokenized text, and the fifth and last module extracts all the results and presents them to the human evaluators. See figure 1.

1. **Data Selection.** N (=100) protein names are combined with M (=37) verb templates, giving a total of N x M (3700) queries to run against Google.
2. **Google.** The queries are fed to the PyGoogle module which allows 1000 queries to be run against the Google search engine every day with a personal password key. In order to maximize the use of this quota, the results of every query are cached locally, so that each given query will be executed only once. If a search returns more than ten results, the resultset can be expanded by ten at a time, at the cost of one of the 1000 quota-queries every time. We decided to use up to 30 results for each query in this experiment.
3. **Tokenization.** The text is tokenized to split it into meaningful tokens, or "words". We use a simple WhiteSpaceTokenizer from NLTK, where every special character (like ( ) " ' - , and .) is treated as a separate token.
4. **Parsing.** Each returned hit from Google contains a "snippet" with the given query phrase and approximately ten words on each side of it. We use some simple regular grammars to match the phrase and the words following it. If the next word is a noun it is returned. Otherwise, adjectives are skipped until a noun is encountered, or a "miss" is returned.
5. **Expert Evaluation.** The results were merged so that all synonyms were treated as if the main protein name had been used in the original query. Then the results were put into groups (one group for each protein-verb pair) and sorted alphabetically within that group. These results were then presented to the biologists, who evaluated the usefulness of our results from Google.

4 Empirical Results

Fig. 2 and 3 show the results. The first one shows that more than half of the extracted terms were terms that could be used to annotate the given protein

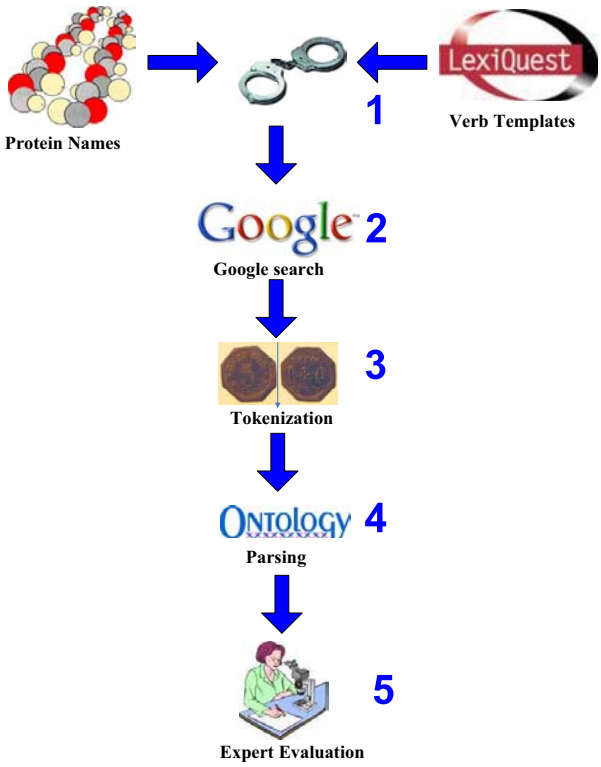

Fig. 1. Overview of Our Approach (named gProt)

according to the Gene Ontology (GO). Around one fifth of the results contained an identifiable protein name that could be stored as a protein-protein interaction. Only one quarter of the terms were deemed not useful. The different kinds of "not useful"-errors can be read out of fig. 3.

5 Related Work

Our specific approach was on using Google and Gene Ontology for annotating protein interactions. We haven't been able to find other work that does this, but the closest are Dingare et al., that uses results from Google search as a feature for a maximum entropy classifier used to detect protein and gene names [5, 6], and our previous work on semantic annotation of proteins (i.e. tagging of individual proteins, not their GO relation) [16]. Google has also been used for semantic tagging outside of the biomedical field, e.g. in Cimiano and Staab's PANKOW system [2] and in [4, 7, 10, 11, 13, 17].

A comprehensive overview of past methods for protein-related information extraction is provided in [18].

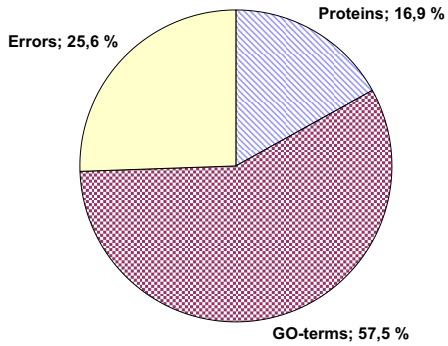

Fig. 2. Main Results

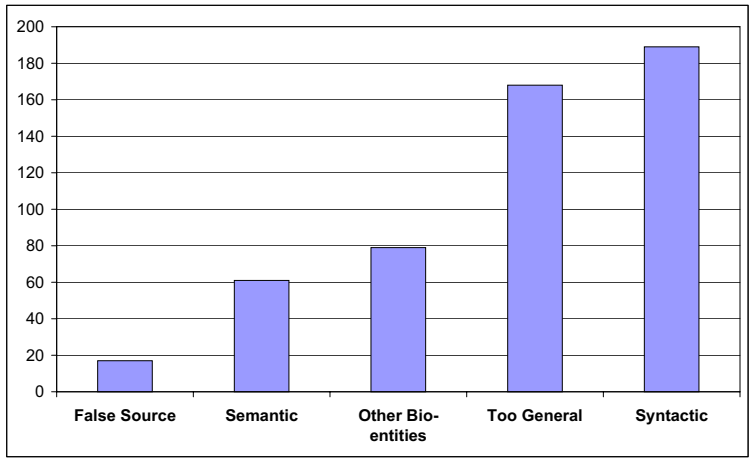

Fig. 3. Breakdown of Errors

6 Discussion

In the following section we discuss our approach step-by-step. (The steps as presented in fig. 1.)

1. **Data Selection.** The results were inspected by cancer researchers, so the focus was naturally on proteins with a role in cancer development, and more specifically cancer in the stomach. One such protein is gastrin, used as a running example in this article. In the experiment we used ten such protein names with around ten synonyms for each. The large number of synonyms used for each original protein name gave us a valuable increase in the recall of expected facts from Google.

2. **Google.** Since we decided to download up to 3 (times 10) results for each query, we had to do around 11.000 queries. This took almost two weeks, because of Google's restraint to only run 1000 queries per day. If we want to scale up this method in the future, we would probably have to pay Google to let us do more queries per day, or consider using the recently announced Yahoo API that allows 5.000 queries per day. The number of returned GO-processes was over 50%, which is very promising for automatic annotation, considering that no information has been used in the process to match GO-terms more often than e.g. protein names.
3. **Tokenization.** Most of the "errors" are syntactic errors, and many of the syntactic errors occur because of bad tokenization, mainly because a lot of the returned words are just parts of multi-word-tokens. Also, many of the words are not nouns at all, so they are not suitable class names in the first place. In the future more work should be done in the tokenization phase. The WhiteSpaceTokenizer was used because it is easy and fast, but with some sort of NP-clustering and parentheses handling, almost half of the errors could be removed. One example of NP clustering is protein names, such as "g-protein coupled receptor (GPCR)".
   How to deal with parentheses? Sometimes they are important parts of a protein name (often part of "formulae" describing the protein), and other times they are just used to state that the words within them aren't that important. And the worst problem is that they are quite often "unbalanced", either because of typing errors, "1) 2) 3)"-style numbering, or smileys.
4. **Parsing.** We used a really simple grammar to extract the interacting terms from what Google returned. It can be summed up as: After the template, keep reading words until a "stop-word" is encountered. As "stop-words" we used some common prepositions, in addition to full-stop punctuation (.,;?!). There is obviously room for a lot of improvements here, e.g. using more advanced Natural Language Understanding techniques.
5. **Expert Evaluation.** The evaluation was quite simple, just focusing on deciding whether this way of using Google to do information extraction is worth pursuing or not. Since the tokenization and grammar modules aren't perfect yet, the biologist also had access to the complete snippets (and the corresponding homepage) in their evaluation work. It is now obvious to us that we should keep developing this system, since almost three out of four results were relevant, and many of them also novel, information.

7 Conclusion and Future Work

This paper presents a novel approach - gProt - using Google to find semantic (GO-) annotations for specific proteins.

We got empirically promising results - 57.5% semantic annotation classes, and 16.9% protein names in the answers provided by gProt. This means that 74.4% of the results are useful. This encourages further work, possibly in combination with other approaches (e.g. rule based information extraction methods), in order

to improve the overall accuracy. In the similar task of protein name identification, recently presented precision scores ranges from 70 to 75% [1]. Hopefully, more advanced methods will greatly reduce the number of errors (useless information), which is currently at 25.6%. Disambiguation is another issue that needs to be further investigated, because sometimes different search-results are really just one single identity, because of synonyms and acronyms for example. Other opportunities for future work include:

- Improve tokenization. Just splitting on whitespace and punctuation characters is *not* good enough. In biomedical texts non-alphabetic characters such as brackets and dashes need to be handled better.
- Search for other verb templates using Google. E.g. Which templates give the best results, and what about negations ("does not activate ...")?
- Investigate whether the Google ranking is correlated with the accuracy of the proposed semantic tag. Are highly ranked pages better sources than lower ranked ones?
- Test our approach on larger datasets, e.g. using *all* the returned results from Google.
- Combine this approach with more advanced natural language parsing techniques in order to improve the accuracy, [14, 15].
- In order to find multiword tokens, one could extend the search query *("X activates")* to also include neighboring words of X, and then see how this affects the number of hits returned by Google. If there is no reduction in the number of hits, this means that the words are "always" printed together and are likely constituents in a multiword token. If you have only one actual hit to begin with, the certainty of the previous statement is of course very weak, but with increasing number of hits, the confidence is also growing.
- In this experiment very crude Part Of Speech (POS) tagging is done, so our results can be seen as a baseline for this kind of experiment. In the future we want to improve the results, for example by utilizing better grammars, and more advanced natural language understanding techniques.

Acknowledgements

We would like to thank Waclaw Kusnierczyk and Tore Amble for continuous support.

References

1. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents (Forthcoming)*, 2004.
2. Philipp Cimiano and Steffen Staab. Learning by Googling. *SIGKDD Explorations Newsletter*, 6(2):24–34, December 2004.

3. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.
4. Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, pages 178–186. ACM, 2003.
5. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, and Beatrice Alex. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. In *Proceedings of the BioCreative Workshop*, March 2004.
6. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. Submitted to BMC Bioinformatics, 2004.
7. Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. Submitted to Artificial Intelligence, 2004.
8. Jun ichi Tsuji and Limsoon Wong. Natural Language Processing and Information Extraction in Biology. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 372–373, 2001.
9. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
10. Vinay Kakade and Madhura Sharangpani. Improving the Precision of Web Search for Medical Domain using Automatic Query Expansion. Online, 2004.
11. Udo Kruschwitz. Automatically Acquired Domain Knowledge for ad hoc Search: Evaluation Results. In *Proceedings of the 2003 Intl. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*. IEEE, 2003.
12. Eric P. G. Martin, Eric G. Bremer, Marie-Claude Guerin, Catherine DeSesa, and Olivier Jouve. Analysis of Protein/Protein Interactions Through Biomedical Literature: Text Mining of Abstracts vs. Text Mining of Full Text Articles. In *Proceedings of the Knowledge Exploration in Life Science Informatics (KELSI2004) Symposium*, volume 3303 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 96–108. Springer-Verlag Heidelberg, 2004.
13. David Parry. A fuzzy ontology for medical document retrieval. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32*, pages 121–126. ACM Press, 2004.
14. Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.
15. Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU Munchen, Germany, April 2003.
16. Rune Sætre, Amund Tveit, Tonje Strœmmen Steigedal, and Astrid Lægreid. Semantic Annotation of Biomedical Literature using Google. In Dr. Marina Gavrilova, Dr. Youngsong Mun, Dr. David Taniar, Dr. Osvaldo Gervasi, Dr. Kenneth Tan, and Dr. Vipin Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO2005)*, Lecture Notes in Computer Science (LNCS) (Forthcoming), Singapore, May 2005. Springer-Verlag Heidelberg.
17. Urvi Shah, Tim Finin, and Anupam Joshi. Information Retrieval on the Semantic Web. In *Proceedings of CIKM 2002*, pages 461–468. ACM Press, 2002.

18. Hagit Shatkay and Ronen Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.
19. Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from Biomedical Literature. In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, Tokyo, Japan, April 2005. IEEE Press (Forthcoming).
20. Limsoon Wong. A Protein Interaction Extraction System. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 520–530, 2001.
21. Limsoon Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.
22. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Paper V

WebProt: Online Mining and Annotation of Biomedical Literature Using Google.
Rune Sætre, Martin T. Ranang, Tonje S. Steigedal, Kamilla Stunes, Kristine Misund, Liv Thommesen and Astrid Lægreid.
Submitted to *Advanced Computational Methods for Biocomputing and Bioimaging.*
Editors: Tuan D. Pham, Hong Yan, Denis I. Crane.
Nova Science Publishers.
Hauppauge, New York 11788-3619.
URL: http://www.novapublishers.com

Chapter 1

WebProt: Online Mining and Annotation of Biomedical Literature Using Google

Rune Sætre [†], Martin T. Ranang [†], Tonje S. Steigedal[‡], Kamilla Stunes[‡], Kristine Misund[‡], Liv Thommesen[‡], Astrid Lægreid [‡]

[†] *Department of Computer and Information Science, Norwegian University of Science and Technology, 7491 Trondheim, Norway*

E-mail: satre@idi.ntnu.no[1], mtr@idi.ntnu.no

[‡] *Department of Cancer and Molecular Medicine, Norwegian University of Science and Technology, 7491 Trondheim, Norway*

E-mail: {tonje.strommenm, kamilla.stunes, kristine.misund, liv.thommesen, astrid.lagreid} @medisin.ntnu.no

[1]to whom correspondence should be addressed

Abstract

Motivation: WebProt is an open source software tool for text mining in molecular biology texts. It is used to collect background information about genes and proteins from online literature sources. This is useful for molecular biologists working with many unfamiliar genes, like for example in a big microarray experiment.
Results: A study using 42 different proteins and their official synonyms/aliases showed that 69,5% of all the results from Google were useful, i.e. containing either *related protein names* or *biological functions, locations and processes* related to the query protein. This is 10% lower than in a previous experiment, so we made a filter requiring each web page to yield at least 6 results, before being taken into consideration. This increased the precision to 91,5%, but the recall dropped to $\frac{2}{3}$, thereby lowering the F-measure of the system from 82% to 78%.
Availability: WebProt is available as a web service system running on http://www.idi.ntnu.no/∼satre/webprot/. It can also be freely downloaded for academic purposes by sending a request to the first author of this chapter. The advantage of using the web based system is that you get access to all the results submitted and searched for by other biologists, and this will speed up some of your searches tremendously.

Keywords: Biomedical Literature Data Mining, Gene Ontology, Google API

1.1 Introduction

In recent years, the interest in developing effective tools for Natural Language Processing (NLP) tasks in biomedical literature has been increasing. There is a practical need to effectively curate, organize and retrieve information automatically from textual sources, and most of these sources have already been indexed by the worlds largest search engines, like Google[2] and Yahoo[3]. With the recent release of Application Programming Interfaces (APIs) to these search engines, a world of new possibilities for practical applications has opened.

This paper describes such a new application, called WebProt. It is an online version of the GProt system that was built with the Google API [15]. It provides a way of automatically updating protein and gene databases by

[2]http://www.google.com/
[3]http://www.yahoo.com/

extracting information from biomedical research literature [22, 21, 19]. This literature is indexed in MEDLINE[4] online, and therefore also by Google and other major search engines. Examples of databases that can be enriched with this kind of automatically extracted information are LocusLink, UniGene and Swiss-Prot for proteins and the Gene Ontology for semantic labels.

The large and rapidly growing amount of biomedical literature demands that the extraction process has to be more *automatic* than previously. Current extraction approaches have provided promising results, but they are not sufficiently accurate and scalable. Methodologically all the suggested approaches belong to the *information extraction field* [3], and in the biomedical domain they range from simple automatic methods to more sophisticated, but then also slightly more manual, methods. Good examples are: Learning relationships between proteins/genes based on co-occurrences in MEDLINE abstracts [9], *manually* developed information extraction rules [23], information extraction classifiers for protein names trained on *manually* annotated training corpora [1], and classifiers trained on *automatically* annotated training corpora [20].

1.1.1 Research Hypothesis

Internet Search Engines such as Google, Yahoo and MSN Search are the largest readily available information sources in the world today, also in the biomedical domain. Based on promising results from recent work on using Google for semantic annotation of biomedical literature [16, 15], we are encouraged to further investigate how Google can be used to find protein interactions that match the Gene Ontology (GO) terms. The hypothesis we are working with is:

Internet Search engines such as Google can be used to automatically annotate protein interactions using the Gene Ontology framework.

1.1.2 Chapter Structure

The rest of this chapter is organized as follows. Section 1.2 describes the materials used, Section 1.3 presents our methods, Section 1.4 presents some empirical results, Section 1.5 describes related work, Section 1.6 discusses our approach and the results we got, Section 1.7 contains some concluding remarks, and finally some ideas for future work are presented in Section 1.8.

[4] http://www.ncbi.nlm.nih.gov/entrez/

1.2 Material

Figure 1.1 shows an overview of the entire system. This section will explain what input we got from the biologists (protein names), and what information we got from public databases and research papers (protein synonyms and verb templates). We will also explain how the data from Google was obtained, and what programming tools and software we used to process the data.

1.2.1 System Input

Table 1.1 contains a list of all the proteins that were used in this experiment. The protein list from the biologists has to be in one of the four standard unique identifier formats made by Entrez Gene, SwissProt, UniGene or GeneBank. In this case, Entrez Gene Identifiers (IDs) were used. The IDs are automatically expanded into a list of all known synonyms by searching through a local copy of the four databases just mentioned.

All the results from Google regarding the proteins in Table 1.1 have now been stored locally in the WebProt system, so if anyone wants to recreate the experiment, the search time will be minimal. Also, since the results are now manually controlled by experts, they can be used as a corpus for future experiments in this kind of Information Extraction domain. All the results are therefore made publicly available at the system web page[5].

Figure 1.2 shows a snapshot of the User Interface (UI) for the WebProt system. It shows protein names and their synonyms on the left hand side, and verbs describing the different interactions on the right hand side. In addition, file upload or manual addition of proteins and synonyms can be done by using the appropriate text boxes and buttons in the protein list. After selecting which proteins, synonyms and verbs to search for, the user pushes the "Search Google" button, and then monitors the search, or just waits until the results are ready. This procedure is explained in more detail in the WebProt Manual, Subsection 1.3.1.

1.2.2 Programming Software

The program is written in the Python programming language[6], using the Google API[7]. The Internet web interface was built using Irmen de Jong's

[5] http://www.idi.ntnu.no/∼satre/webprot/
[6] http://www.python.org/
[7] http://www.google.com/apis/

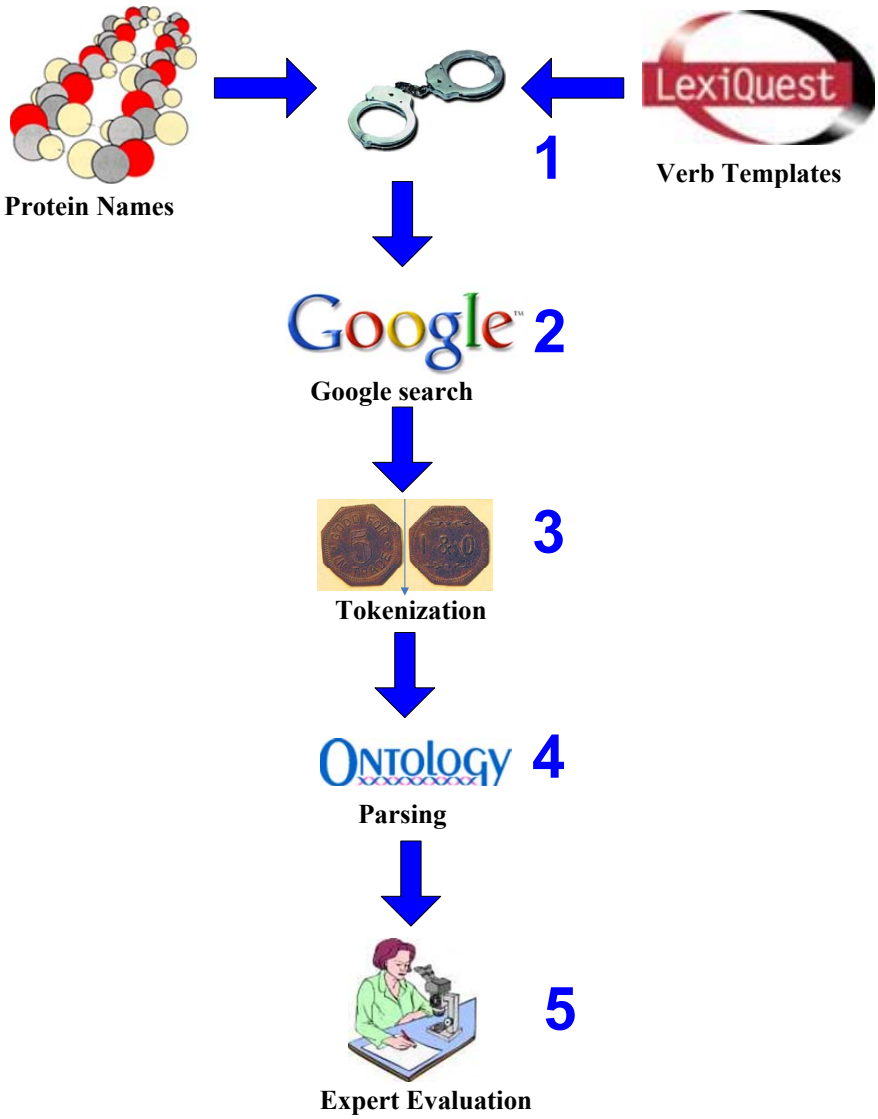

Figure 1.1: Data flow in the WebProt System.

Misund facts: 1118, bad:54%			Thommesen facts: 492, bad:17%		
SYMBOL (ID)	Facts	Bad	SYMBOL (ID)	Facts	Bad
BMP2 (650)	69	0%	CREBBP (1387)	81	74%
BMP3 (651)	1	0%	FHL2 (2274)	44	2%
CCNA2 (890)	81	9%	KCNA2 (3737)	56	11%
CHGB (1114)	1	0%	PLA2G2A(5320)	59	3%
CREM (1390)	85	6%	PRKD1 (5587)	71	15%
ELK1 (2002)	10	0%	RPS6KA1(6195)	74	0%
MAP3K4 (4216)	8	0%	RPS6KA2(6196)	49	0%
PAX6 (5080)	662	89%	AKAP1 (8165)	9	0%
SRF (6722)	116	5%	BTAF1 (9044)	13	0%
RPS6KA5 (9252)	27	0%	RPS6KA5(9252)	27	0%
ICER (378903)	58	5%	TORC2 (200186)	9	44%
Stunes facts: 2582, bad:20%			Steigedal facts: 472, bad:44%		
SYMBOL (ID)	Facts	Bad	SYMBOL (ID)	Facts	Bad
BMP1 (649)	210	18%	CCND1 (595)	15	0%
RUNX2 (860)	69	0%	CHGA (1113)	51	22%
COL1A1 (1277)	2	0%	EGR3 (1960)	7	0%
IL6 (3569)	927	1%	FGF1 (2246)	53	2%
OPG (4982)	132	5%	FOS (2353)	126	37%
SLC6A4 (6532)	40	3%	GATA6 (2627)	9	0%
SST (6750)	428	28%	HDC (3067)	24	88%
TPH1 (7166)	8	100%	NP (4860)	180	71%
TNFSF11 (8600)	241	29%	CCNI (10983)	3	0%
TNFRSF11A (8792)	269	78%			
RPS6KA5 (9252)	27	0%			
GHRL (51738)	66	85%			
SP7 (121340)	163	0%			

Table 1.1: Proteins and corresponding Entrez Gene ID numbers for the experiment. Number of facts found by Google, and error-rate is also shown.

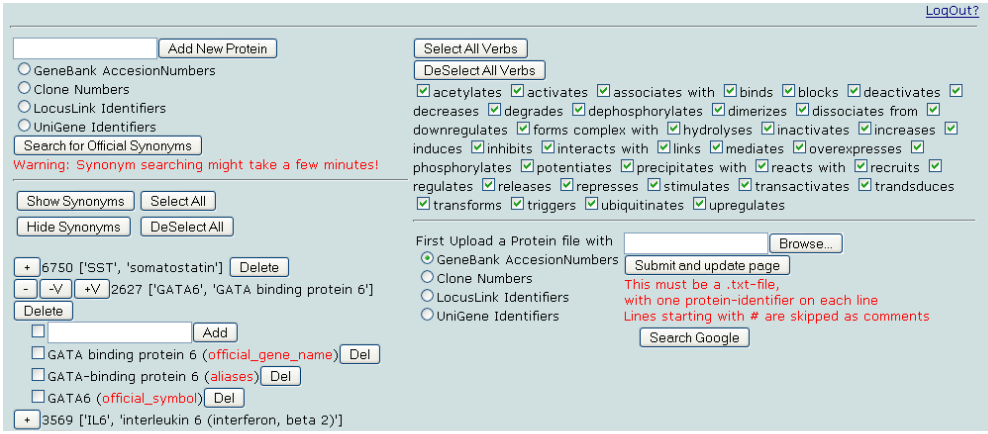

Figure 1.2: WebProt User Interface

Snakelets[8].

1.3 Methods

The goal of our experiment was to show that Google is a good source to use for extracting information about gene and protein interactions. To show this, we built a real online web service, allowing biologists to search for their favourite proteins, and then giving us their opinion about how useful the results were, and how many errors they found. The content of this section is:

- A short WebProt User Manual (1.3.1)
- Overview of the experimental setup and result classification (1.3.2)
- More details on the Google Interface (1.3.3)
- A description of the Natural Language Processing (NLP) part of the system (1.3.4)
- Information about the automatic Gene Ontology Annotation algorithm (1.3.5)

[8]http://snakelets.sourceforge.net/

1.3.1 Short WebProt User Manual

This subsection explains how to use the WebProt system, and also what you need to do to make a local copy of the system.

Since the WebProt system is based on the Google API, you need a *Google Key*[9] to run your queries. In order to get a Google Key, you first need a Google user account, which you will be prompted to create during your first login to the WebProt system. The Google Key will be stored permanently in the system, and it will automatically be used to perform all your following queries.

After logging in, you will see the main WebProt page, as shown in Figure 1.2. When using the system for the first time, your personal protein list will be empty, so you have two choices:

1. You can upload a file with Protein Identifiers in one of the four allowed formats (UniGene, SwissProt, Entrez Gene or GeneBank)

2. You can manually add one identifier at the time, using one of the four different ID types for each entry. You can also "invent" your own protein identifiers, but then the automatic synonym expansion will not work for those identifiers.

When you have registered the proteins that you want to study, you can use the program to automatically find all official synonyms and symbols for the identifiers that you have entered. This is done by indicating what ID format you have used, and clicking the "Get all official synonyms" button. Finally, when all protein and synonyms are present, you can choose from a standard set of interaction verbs [12] and push the "Search Google" button to start your search. Google only allows 1000 queries per day per user, so if your search contains more than 1000 queries, you have to restart it the next day. All the results will be stored in the WebProt system, so the search will automatically continue from where it stopped the last day. Since our local copy of the results from Google is a shared resource for all the users of the system, you will never have to use your quota to do a query that has been done by you or any other user previously. To make sure that the copies of the results do not get to old, we mark every result with a date stamp, saying when the search was done. In this way, we can delete or omit results that are older than a certain threshold.

After the search phase, the evaluation phase follows. All the results are listed, sorted by protein and interaction verb, and there are clickable links

[9] http://www.google.com/apis/

Result	Label	Explanation
Correct	*Prot*	Proper gene or protein term
	GO	Proper Gene Ontology (GO) term
Bad NLP	*SnipProt*	Unextracted protein/gene in snippet
	SnipGO	Unextracted GO term in snippet
Bad results	*Bad*	No (correct) facts in snippet

Table 1.2: Result Classes

into the web pages containing the results, and into the Amigo GeneOntology Browser for GO terms.

If you want to install the WebProt system on your own web server, you first have to download and install Python and Snakelets. Then you can copy the latest version of the WebProt files[10] into the Snakelets webapps directory and change the __init__.py file to include the WebProt system as one of your web applications. If you do this, you will not have the benefit of accessing results from other biologists, but it will give you more freedom in changing the program to do exactly what you want.

1.3.2 Experiment Configurations

After the user has submitted the protein list, the synonyms are automatically extracted from biological databases, and the Google search is done as described in the last subsection. Then we have enough information to answer the following two questions:

1. How much relevant information (e.g. protein names and GeneOntology codes) do the snippets from Google contain?

2. How easy is it to automatically extract the exact protein names or GO codes from the snippet, using just very simple Natural Language Processing techniques?

We used four positive and one negative label to classify the results (Table 1.2). The *SnipProt* and *SnipGO* tags means that a correct humanly readable fact is present in the snippet text from Google, but the fact was not successfully extracted by the system because of erroneous NLP processing.

[10] satre@idi.ntnu.no

1.3.3 Google Searching

Since each user can only do 1000 queries per day, queries are the scarcest resource in the system now. Also, a query returning no results have the same "cost" as a query returning a full Google result set with useful results. Currently, each of the 1000 result sets can only contain a maximum of 10 single results, so if we want to see the first 30 hits for a query, we have to do three separate searches for this query. This also means that if you can find a clever way to avoid the empty queries, you can download almost 10.000 useful results per day, instead of just 1000 empty ones.

Our queries to Google are quite specific, including quotes (" query ") to indicate that only exact matches should be accepted, and this leads to many empty answers from Google. Even without the quotes, many synonym and verb combination show up as empty searches, simply because many of the official synonyms are hardly ever used in written articles or abstracts. So, to avoid wasting one query for every verb on a single unused synonym, the search algorithm was implemented in the following way:

1. First, search for the individual synonym alone, without any accompanying verbs.

2. If Google estimated the size of the entire result set to be lower than 500 hits (50 queries), then download all the results, and skip the next step.

3. If there are more than 500 estimated hits for a query, make a single query for every synonym and verb combination, leading to a maximum of (40verbs x 3queries =) 120 queries.

The advantage of this method is that it avoids wasting queries in the case of rare synonyms. At the same time, it guarantees that all the results for a given " synonym verb " query contains interesting sentence patterns, and it will only use as many queries for that query as there are actually matching sentences on the Web.

A similar problem of wasting queries occurs when a protein synonym is a very common word, like "AN", "AND" or "GAS". See PAX6 and TPH1 in Table 1.1 for a real example. In this case, Google will find many results to download, but there is a good chance that none of them will be related to molecular biology, so we will have wasted our quota again. This also introduces a lot of extra errors into the extracted data, as long as more powerful NLP techniques are not used. The only way to avoid this problem currently, is for the user to remove some of the bad automatically generated synonyms, before starting the search.

Stopword	Explanation
...	End of snippet, indicating that something is missing...
.	End of the sentence (usually), ok
;	Semicolon, ok
and	Conjunction, indicating that one more fact following
as	Adverb/Conjunction, indicating new clause
but	Conjunction, indicating doubt or negative fact following
is	Verb, indicating passive sentence structure
or	Conjunction, indicating doubt or more facts following
while	Conjunction, ok

Table 1.3: Stopwords, marking the end of potential protein names.

1.3.4 The Natural Language Processing (NLP) part

The results returned from Google include a field called *snippet*, containing the query and its context. Each snippet usually consists of one or two sentences, including the exact phrase that we searched for (approximately ten words on each side of the query). The snippet is extracted from the corresponding web page, which can be in a lot of different formats, e.g. HTML, Word Document, PowerPoint Presentation slides, PDF file and so on. In the snippets, (almost) all original formatting codes like HTML or bullet list characters have been removed by Google. This leaves just plain text, which would be an ideal target for NLP, but during the search, Google automatically adds some HTML code to the snippet again, to highlight the actual query terms. This makes it very easy to read the results when viewing them online, but in the case of automatic text processing, HTML codes are just noise that has to be removed before applying the text matching algorithm.

After removing the HTML codes we try to retrieve our query from the snippet, but sometimes the snippet does not contain the exact phrase that we searched for. This can happen for two reasons: First, when Google build their search index they ignore several punctuation characters that are actually often important parts of protein or gene sequence names. Second, if a large number of people create links to a web page using for example a protein name as the link term, Google will take this into account, and return the linked page, even if it does not actually contain the given search query. If our query is not present in the snippet, we skip to the next result, assuming that the current one is garbage.

If the snippet contains the exact search phrase, we remove the phrase and everything before it. Then we store all the following words until we encounter

Stopword Prepositions
against, between, by, despite, for, from, in, into,
#of, on, than, #that, through, to, towards, upstream,
until, using, versus, via, which, whose, with, within, without

Table 1.4: Prepositions, marking the end of potential protein names.

one of the special *stopwords* in Table 1.3, or one of the prepositions in Table 1.4. We removed *"of"* and *"that"* from this list, since they are often part of interesting general protein descriptions like "a gene *that* activates this protein".

After the candidate protein and GO names have been extracted like this, we remove all occurrences of words like *"the"* to reduce the number of double hits, i.e. we do not want to distinguish between "ACL protein" and "the ACL protein". There is obviously room for a lot of improvements here, e.g. using more advanced Natural Language Understanding techniques, but at least this simple method ensures good coverage.

Before the resulting terms from this algorithm are sent back to the user for manual evaluation of the quality, a fast algorithm extracts all possible GO candidates from the (full) snippet text. This makes it easier for the users who can not call to mind all of the approximately 20.000 terms (plus synonyms) in the Gene Ontology. The GO matching algorithm is described in the next subsection.

1.3.5 Automatic Gene Ontology Annotation

To help the human experts evaluate the results, every snippet was automatically annotated with a set of all the Gene Ontology (GO) identifiers and terms (or term synonyms) that appeared in the snippet. See Algorithm 1 for a pseudocode description of the GO matching algorithm.

The algorithm takes three arguments as its input. The first argument is a list, denoted T_{GO}, which represents the information extracted from the file *gene_ontology.obo*[11]. Each element of T_{GO} is a tuple, $(i_{\mathrm{GO}}, t_{\mathrm{GO}})$, where i_{GO} denotes the GO identifier, and t_{GO} denotes the corresponding tokenized GO term (or synonym). By *tokenized*, we mean that the string representing the term (or synonym) was split into several shorter strings, each representing a token (a single word or a punctuation symbol), and stored as a list.

The second argument is a hash index, denoted I_{GO}. For each element

[11] All Gene Ontology Terms and Synonyms, available at http://www.geneontology.org/ontology/gene_ontology.obo.

Algorithm 1 Identify any GO term (or synonym) occurring in the sequence of tokens T, given a list of tokenized GO terms T_{GO} and a corresponding hash index I_{GO}.

```
1:  function FindTerms(T_GO, I_GO, T)
2:      result ← {}
3:      i ← 0
4:      while i < |T| do
5:          j ← 0                          ▷ Number of matching tokens since i.
6:          span ← 0                ▷ The length of the current match(es) in T.
7:          parts ← {}
8:          trailing_garbage ← 0
9:          done ← False
10:         while ¬done ∧ (i < |T|) do
11:             t ← T[i + span]
12:             if t ∈ ignorable then
13:                 if span = 0 then
14:                     break          ▷ No path starts with an ignorable token.
15:                 else
16:                     span ← span + 1
17:                     trailing_garbage ← trailing_garbage + 1
18:                     continue
19:             key ← (j, t)
20:             if key ∈ I_GO then
21:                 if span = 0 then
22:                     parts ← I_GO[key]          ▷ Note that parts is a set.
23:                 else
24:                     parts ← I_GO[key] ∩ parts
25:                 j ← j + 1
26:                 span ← span + 1
27:                 trailing_garbage ← 0                ▷ No trailing garbage.
28:             else
29:                 done ← True
30:             for all k ∈ {n|(n ∈ parts) ∧ (|T_GO[n][1]| = j)} do
31:                 result ← result ∪ {(i, (i + span − trailing_garbage), k)}
32:         i ← i + 1
33:     return result
```

$(i_{\mathrm{GO}}, t_{\mathrm{GO}}) \in T_{\mathrm{GO}}$, several tuples (i, t_i) are generated (t_i is the ith token in t_{GO}). These tuples are used as the *keys* in I_{GO}, while the *value* associated with each key is a set. Each such set contains the indices of the elements in T_{GO} that were used to generate the key identifying it.

The third argument of the algorithm is a tokenized snippet. Every snippet is tokenized just like the GO terms.

The returned value from the algorithm is a set of tuples $\{(n_k, m_k, i_k), \dots\}$, where n_k and m_k are the start and stop indices of the kth match, while i_k is the index of the matching GO term in T_{GO}.

The algorithm is designed to allow certain tokens from a set called *ignorable* to occur within a matching region. Also, note that no distinction is made between upper and lower case characters in the tokens.

1.4 Results

This section summarizes the results from the experiments, together with an estimate of the success rate of the system, calculated as precision, recall and F-measure values. A table showing the top ten contributing web domains giving this kind of protein interaction facts is also given.

1.4.1 Precision, Recall and F-measure

After the experiments, the performance of the system was computed according to (1.1), (1.2) and (1.3)

$$(P)recision = \frac{|\text{correct facts}|}{|\text{all extracted facts}|} = \frac{3243}{4606} = 70\% \quad (1.1)$$

$$(R)ecall/Coverage = \frac{|\text{correct facts}|}{|\text{all relevant facts}|} = \frac{3243}{3243?} \leq 100\% \quad (1.2)$$

$$\text{F-measure} = \frac{2*P*R}{P+R} = \frac{2 * 0.7 * 1}{0.7 + 1} = 82.4\% \quad (1.3)$$

The question mark in formula 1.2 shows that we do not have the exact number of possible facts, since nobody knows how much relevant data is actually stored on the internet. One way to get around this problem, would be to let a biologist create a set of certain facts for a limited number of proteins, and then see how many of these facts are actually found by the system. This will at least give us an estimate of the coverage of the system, but there is

still a chance that the system can find new data, previously unknown even to the biologist. Also, creating a good gold standard like this is too expensive, so in this experiment we use the system to help us build the gold standard file, by measuring the precision manually after running the experiments, and calculating the F-measure by assuming that all the positive facts were found by the system during the search. This is a farfetched assumption, but at least we did not see any counter examples (yet). Table 1.5 shows how precision, recall and F-measure values change, when different values for the hit-count filter are used. A hit-count filter with value three, means that only web domains contributing three or more different facts to the search are listed at valid results.

Table 1.6 gives a list of the web domains that contributed most facts in response to the queries. As you can see, almost all the major contributors are electronic (online) journals. One exception to this is uam.es[12], which is hosting a system called iHOP [8]. This system has extracted half a million sentences from MEDLINE and is using HTML markup to allow the user to navigate between research articles just by clicking on protein names in the text. The biggest contributor of facts is, not surprisingly, the National Library of Medicine[13], since they have the MEDLINE collection of 12 million abstracts online, in addition to several journal and book articles.

1.4.2 Time Consumption

To determine how fast the system could operate if Google removed the limit of 1000 queries per, the average time it took to do one single query was measured. If you want to do 5000 queries with the Google API right now, it takes five days for a single user to complete the search, because the experiment must be restarted every day after reaching 1000 queries. If we need the results faster, maybe we could buy a bigger quota from Google?

The actual time used to do just one such Google Search query varies between two and four seconds, depending on network load etc. This was measured by running 1000 queries, which usually took between 40 and 60 minutes to complete. Three seconds per query is rather slow, and the main reason for this is probably that Google tries to limit the bandwidth given to one single user at any time. If you run the same query on the Google web page, the search time (without network latency) is reported to be around 0.2 to 0.4 seconds, which is ten times faster than what the program achieves. Another

[12]http://www.pdg.cnb.uam.es/UniPub/iHOP/

[13]http://www.ncbi.nlm.nih.gov/

Hit limit	Precision	Recall	F-measure	Facts	Correct
1	69,5	100,0 %	82,0 %	4651	3243
2	82,7	84,4 %	83,5 %	3310	2737
3	86,2	77,7 %	81,7 %	2922	2519
4	87,9	73,5 %	80,0 %	2709	2382
5	89,4	70,4 %	78,7 %	2553	2282
6	91,5	67,5 %	77,7 %	2393	2189
7	92,3	64,4 %	75,8 %	2261	2087
8	92,4	63,0 %	74,9 %	2212	2043
9	93,9	61,0 %	74,0 %	2108	1979
10	94,1	60,4 %	73,6 %	2081	1959
11	94,6	59,8 %	73,3 %	2051	1940
12	94,5	58,5 %	72,3 %	2007	1897
20	94,3	54,5 %	69,1 %	1874	1767
30	94,8	46,5 %	62,4 %	1592	1509

Table 1.5: Different values for web domain hit-count limit, leading to different P, R and F-measures.

problem with automatic querying is that Google sometimes does not answer the query at all, and then the Simple Object Application Programming interface (SOAP[14]) returns an error message. Then WebProt has to deal with the problem, for example by running the erroneous query again. This leads to at least a doubling of the search time for the erroneous query, and in the worst case it can lead to inconsistencies in the local database of WebProt, for example if a new type of error occurs, and the system has not been programmed to properly handle it.

1.5 Related Work

Our specific approach was on using Google and Gene Ontology for annotating protein interactions. We have not been able to find any other work that does this, but the closest is Dingare et al., that uses results from Google search as a feature for a maximum entropy classifier used to detect protein and gene names [5, 6].

The results presented in this paper are extensions of our previous work on semantic annotation of proteins [15]. Google has also been used for semantic

[14]http://www.w3.org/TR/soap12-part1/

Domain	Facts	Explanation
nih.gov	239	PubMed Central Collection of Journals, Books and MEDLINE
jbc.org	196	Biological Chemistry, Journal
physiology.org	143	American Physiological Society, Collection of Journals
endojournals.org	110	Endocrine Society, Collection of Journals
asm.org	83	American Society for Microbiology, Collection of Journals
ahajournals.org	71	American Heart Association, Collection of Journals
nature.com	69	Nature, same as npgjournals.com, Collection of Journals
aacrjournals.org	55	Cancer Research Journal
ingentaconnect.com	55	Ingenta Online Publisher, Collection of Journals
jimmunol.org	51	Immunology, Journal
karger.com	48	Karger Medical and Scientific Publisher, Big Collection of Journals
pnas.org	44	National Academy of Sciences USA, Proceedings
ac.uk	42	MOLECULAR AND CELLULAR BIOLOGY, Journal Laboratory of Molecular Biophysics University of Oxford, Journal
bloodjournal.org	40	American Society of Hematology, Blood Journal
uam.es	39	Information Hyperlinked over Proteins (iHOP), Network
aspetjournals.org	38	Molecular Pharmacology Journal Pharmacology and Experimental Therapeutics, The Journal
oxfordjournals.org	33	Cerebral Cortex Journal Nucleic Acids Research Journal Human Molecular Genetics Journal Nephrology Dialysis Transplantation Journal
jcb.org	32	Cell Biology, Journal
blackwell-synergy.com	32	Neurochemistry, Journal
npgjournals.com	30	Collection including European Molecular Biology Organization Journal
biochemj.org	30	Biochemical Journal

Table 1.6: Web domains contributing with 30 facts or more to the results.

tagging outside of the biomedical field, e.g. in Cimiano and Staab's PANKOW system [2] and in other systems [17, 7, 10, 11, 4, 13].

A comprehensive overview of past methods for protein-related information extraction is provided in [18].

1.6 Discussion

This section will present some afterthoughts concerning the quality of the method and the results. Specifically, it will explain what kind of problems the Natural Language Processing (NLP) part of the system encountered, and how the built-in Page Rank feature of Google can be used to evaluate the general quality of the retrieved data.

1.6.1 Error Classification

There are many reasons why the NLP part of the system can give bad results, e.g.:

- Bad conversion from PowerPoint files to HTML text in Google.
- "Invisible" formatting codes from Portable Document Format (PDF) or PostScript (PS) files sometimes find their way into the snippets that are returned from Google.
- Periods and hyphen symbols used for example in bulleted lists in Microsoft Word or PowerPoint files are not treated properly.
- "Merged" characters like "fi" and "fj" because of a typography technique known as ligatures.

The last problem could be solved by our parser by doing a dictionary lookup for example, but it is probably a better approach to ask Google to deal with this on their server side. Then all the possible client applications that will be made around the world can avoid lots of identical error processing.

Another source of bad results from Google is the fact that the snippets are very short, so quite often the long "research language" sentences will be cut in the middle of the interesting fact that we want to extract. This can probably be solved in one of two ways. Since we have the URL to the original document, we could download all the text, and do parsing of the entire document. Unfortunately, this is very time consuming, and it also makes it necessary to

deal with formatted PowerPoint, PDF or other complex file formats. The second way to deal with snippet cut-offs could be to use an extra Google search specifying only the last words just before the cut-off (...) and thereby tricking Google into revealing more of the resulting snippet. This second approach has not been tried in practice yet.

1.6.2 Page Rank

When WebProt uses Google to extract information for a given synonym and verb pair, it only downloads 30 results, even if several hundred results exists. This means that we are relying on the Google Page Rank formula to do much of the filtering for us. Considering how the Page Rank is calculated, this is probably a good thing, since the Page Rank takes into account the popularity of the page, what kind of terms people use to link to it and what terms they use when they search for and then view the page from the Google web site.

1.7 Conclusion

We studied how Google can be used to find biological relevant information on protein interactions from the internet. This was done using a self-made Python program, and the results are very promising. To help you use this in your own research, the program is available both online [15], and as source code by request to the first author of this chapter [16].

There are some weaknesses (or room for improvement), especially in the automatic determination of the correctness of retrieved results, which is why the biologists still need to be in the loop. Also, a study should be done to determine how complete the results are, but this requires a gold standard corpus for this kind of information retrieval. Building such a corpus is extremely time consuming, so more manpower is needed, but it should be possible to extend the current results into such a corpus for others to use. See the Future Work Section for more detail on this.

1.8 Future Work

- How can this system be improved further?

[15] http://www.idi.ntnu.no/∼satre/webprot/

[16] satre@idi.ntnu.no

System	Step	Data	Example
WebProt	1	Input:	CCK
	2	→ Google:	” CCK activates ”
	3	Snippet:	CCK activates CREB and ICER.
Bioogle	4	Tagged object:	CREB
	5	→ Google:	” CREB is (an|a) ”
	6	Snippet:	CREB is a protein.
	7	Extracted class:	protein.

Figure 1.3: How WebProt and Bioogle can be combined.

- How can we make it more automatic and reduce the workload of the biologists even more?
- How can we automate the process of telling if extracted facts are really biological entities or not?

The next natural extension to the WebProt program is to do more advanced parsing of the snippets from Google. Then it will be easier to find the object of the verbs, and once the object phrase is known, it is possible to use Google to determine the class that the object belongs to (see Figure 1.3). If the returned class of the object does not match any of the biological classes in our hierarchy, we can assume that the extracted fact was not a relevant one. This is based on the approach taken in the Bioogle system[16]. We are currently developing a molecular biology domain language parser and question-answering system, called GeneTUC [14], which can help us do this kind of subject and object tagging.

In order to ensure that the extracted facts are really biological relevant, it is possible to do more advanced filtering of the web pages that we let Google search. For example, the search could be constricted only to NCBIs MEDLINE pages, but that would probably remove too many very recent facts from being discovery. A better solution is probably to allow all pages in the search, and then gradually build a positive filter listing all the pages like MEDLINE, journals and university websites, that give useful results. When this filter is rich enough, we can finally stop showing pages that are not among these good sources.

Appendix A

Acknowledgements

The first author would like to thank his supervisors Tore Amble, for his continuous support throughout the entire PhD program, and Amund Tveit, for giving me a little push whenever I need it. Also, thanks to professor Tsujii and all the people in his lab in Tokyo University for hosting me during the writing of this chapter. Finally, thanks to Frode Høyvik, who provided valuable input after proofreading this manuscript.

Bibliography

[1] Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.

[2] Philipp Cimiano and Steffen Staab. Learning by Googling. *SIGKDD Explorations Newsletter*, 6(2):24–34, December 2004.

[3] J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.

[4] Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, pages 178–186. ACM, 2003.

[5] Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, and Beatrice Alex. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. In *Proceedings of the BioCreative Workshop*, March 2004.

[6] Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. Submitted to BMC Bioinformatics, 2004.

[7] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Un-

supervised Named-Entity Extraction from the Web: An Experimental Study. Submitted to Artificial Intelligence, 2004.

[8] Robert Hoffmann and Alfonso Valencia. A gene network for navigating the literature. *Nature Genetics*, 36(7):664, May 2004.

[9] Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.

[10] Vinay Kakade and Madhura Sharangpani. Improving the Precision of Web Search for Medical Domain using Automatic Query Expansion. Online, 2004.

[11] Udo Kruschwitz. Automatically Acquired Domain Knowledge for ad hoc Search: Evaluation Results. In *Proceedings of the 2003 Intl. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*. IEEE, 2003.

[12] Eric P. G. Martin, Eric G. Bremer, Marie-Claude Guerin, Catherine DeSesa, and Olivier Jouve. Analysis of protein/protein interactions through biomedical literature: Text mining of abstracts vs. text mining of full text articles. In *Knowledge Exploration in Life Science Informatics (KELSI2004) Symposium*, number 3303 in Lecture Notes in Artificial Intelligence (LNAI), pages 96–108. Springer Verlag, 2004.

[13] David Parry. A fuzzy ontology for medical document retrieval. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32*, pages 121–126. ACM Press, 2004.

[14] Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.

[15] Rune Sætre, Amund Tveit, Martin Thorsen Ranang, Tonje Strømmen Steigedal, Liv Thommesen, Kamilla Stunes, and Astrid Lægreid. GProt: Annotating Protein Interactions Using Google and Gene Ontology. In *Lecture Notes in Computer Science: Proceedings of the Knowledge Based Intelligent Information and Engineering Systems (KES2005)*, volume 3683, pages 1195 – 1203, Melbourne, Australia, August 2005. KES 2005, Springer.

[16] Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal, and Astrid Lægreid. Semantic Annotation of Biomedical Literature using Google. In Dr. Osvaldo Gervasi, Dr. Marina Gavrilova, Dr. Youngsong Mun, Dr. David Taniar, Dr. Kenneth Tan, and Dr. Vipin Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO 2005)*, volume 3482 (Part III) of *Lecture Notes in Computer Science (LNCS)*, pages 327–337, Singapore, May 9-12 2005. Springer-Verlag Heidelberg.

[17] Urvi Shah, Tim Finin, and Anupam Joshi. Information Retrieval on the Semantic Web. In *Proceedings of CIKM 2002*, pages 461–468. ACM Press, 2002.

[18] Hagit Shatkay and Ronen Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.

[19] Jun'ichi Tsujii and Limsoon Wong. Natural Language Processing and Information Extraction in Biology. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 372–373, 2001.

[20] Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from . In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, pages 1161–1161, Tokyo, Japan, April 2005. IEEE Press (Electronic Publication).

[21] Limsoon Wong. A Protein Interaction Extraction System. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 520–530, 2001.

[22] Limsoon Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.

[23] Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Paper VI

GeneTUC, GENIA and Google: Natural Language Understanding in Molecular Biology Literature.
Rune Sætre, Harald Søvik, Tore Amble and Yoshimasa Tsuruoka.
In *Special Issue on "Data Mining and Bioinformatics" of Transactions on Computational Systems Biology*,
Lecture Notes in BIology LNBI, Volume 4070, part V, pages. 68–82.
Editor-in-Chief: Dr. Corrado Priami, University of Trento, Italy.
Springer-Verlag Berlin Heidelberg, Germany, 2006.
ISSN: 0302-9743.
URL: http://www.springer.com/sgw/cda/frontpage/0,,4-164-2-109319-0,00.html

GeneTUC, GENIA and Google: Natural Language Understanding in Molecular Biology Literature

Rune Sætre[1], Harald Søvik[1], Tore Amble[1], and Yoshimasa Tsuruoka[2]

[1] Department of Computer and Information Science,
Norwegian University of Science and Technology,
Sem Sælandsv. 7-9, NO-7491 Trondheim, Norway
Rune.Satre@idi.ntnu.no
http://www.idi.ntnu.no/~satre
[2] Department of Computer Science, University of Tokyo,
Hongo 7-3-1, Bunkyo-ku, Tokyo 113-0033, Japan

Abstract. With the increasing amount of biomedical literature, there is a need for automatic extraction of information to support biomedical researchers. GeneTUC has been developed to be able to read biological texts and answer questions about them afterwards. The knowledge base of the system is constructed by parsing MEDLINE abstracts or other online text strings retrieved by the Google API. When the system encounters words that are not in the dictionary, the Google API can be used to automatically determine the semantic class of the word and add it to the dictionary. The performance of the GeneTUC parser was tested and compared to the manually tagged GENIA corpus with EvalB, giving bracketing precision and recall scores of 70,6% and 53,9% respectively. GeneTUC was able to parse 60,2% of the sentences, and the POS-tagging accuracy was 86.0%. This is not as high as the best taggers and parsers available, but GeneTUC is also capable of doing deep reasoning, like anaphora resolution and question answering, which is not a part of the state-of-the-art parsers.

Keywords: Biomedical Literature Data Mining, Google API, GENIA.

1 Introduction

Modern research is presenting more new and exciting results than ever before, and it is gradually becoming impossible for the human reader to stay up-to-date in the sea of information. This is especially true in the Medical and Molecular Biology domains, where the MEDLINE database of publications is increasing with almost 2000 new entries every day. To help researchers find the information they are searching for in an efficient manner, automatic Information Extraction (IE) is needed. This paper describes a system that is using Natural Language Processing (NLP) in order to automatically read the abstracts of research papers, and later answer questions posed in English about the abstracts.

C. Priami et al. (Eds.): Trans. on Comput. Syst. Biol. V, LNBI 4070, pp. 68–82, 2006.

1.1 Information Extraction (IE) in Biology

The large and rapidly growing amounts of biomedical literature demands a more *automatic* extraction process than previously. Current extraction approaches have provided promising results, but they are not sufficiently accurate and scalable. A survey describing different approaches within the *information extraction field* is presented in [6], and a more recent "IE in Biology" survey is given in [15]. In the biomedical domain, IE approaches range from simple automatic methods to more sophisticated but also more manual methods. Some good examples are: Learning relationships between proteins/genes based on co-occurrences in MEDLINE abstracts [9], *manually* developed IE rules [24], protein name classifiers trained on *manually* annotated training corpora [1], and classifiers trained on *automatically* annotated training corpora [20].

A new emerging approach to medical IE is the heavy use of corpora. The workload can then be shifted from the extremely time consuming manual grammar construction to the somewhat easier and more teamwork oriented corpus/treebank building [12]. This means that the information acquisition bottleneck can be overcome, while still reaching state-of-the-art coverage scores (around 70-80 percent). In this chapter a corpus is used, namely the GENIA Tree Bank (GTB) corpus [19], first to train and then later to test how well the GeneTUC parser performs compared to other available parsers in this domain.

1.2 GeneTUC

The application that we want to improve and test, by incorporating alternative sources of information, is called GeneTUC. TUC is short for "The Understanding Computer", and is a system that is under continuous development at the Norwegian University of Science and Technology. Section 3 will explain in more detail how TUC, and especially GeneTUC, works.

1.3 Chapter Structure

The rest of this chapter is organized as follows. Section 2 describes the materials and programs that were used, section 3 explains in detail how GeneTUC works, section 4 presents our approach, section 5 presents the empirical results, section 6 describes other related work, section 7 contains a discussion of the results, and finally the conclusion and future work are presented in section 8.

2 Materials

One of the main goals was to test how good the current state of the GeneTUC parser is. To do this, some manually inspected parsed text is needed, and that is exactly what the new syntactically enhanced GENIA corpus is [19]. It consists of text from MEDLINE (see subsection 2.1), and provides a gold standard that can be used both for training and testing the GeneTUC application. See Subsection 2.2 for more details.

2.1 MEDLINE

Medline[1] is an online collection of more than 14 million abstracts by now (November 2005). The abstracts are collected from a set of different medical journals by the US National Institutes of Health (NIH). NIH grants academic licenses for PubMed/MEDLINE for free to anyone interested. When it was downloaded in September 2004, the academic package included a local copy of 6.7 million abstracts, out of the 12.6 million entries that were available on their web interface at that time.

2.2 GENIA Tree Bank (GTB)

It was decided to use the GENIA Tree-Bank (GTB) corpus[2] for training and testing of GeneTUC. GTB consists of 500 abstracts from the GENIA corpus which consists of 2000 abstracts from MEDLINE. These 500 abstracts have been parsed, manually inspected and corrected to ensure that they contain the expected parse result for every single sentence. The format of the annotation is a slightly modified Penn Tree Bank-II format. The GTB is split into GTB200 with 200 abstracts and GTB300 with 300 abstracts. We used GTB300 as a training set, and GTB200 as test set to calculate the precision and recall scores for parsing of *unseen* text. It should be pointed out that GTB is still in a beta-release state, which means that it still contains some errors, and some manual inspection of the results are needed to determine if this has a great influence on the measured numbers.

A list of all composite terms in the GTB was also used as input to the system. This was done to ensure that the parsing performance was measured without being influenced by bad tokenization, which is handled by another module, namely the lexical analysis module, in GeneTUC.

3 GeneTUC

GeneTUC is on the way to be a full-fledged Question Answering system, but the coverage is still low. Figure 1 shows the general information flow in the TUC systems. The input sentence can be either a fact for example from a Medline abstract or a question from the user. The analysis is the same in both cases, but the answer will have two different forms. In the case of a factual input sentence, the facts are coded in a first order event logic form called TUC Query Language (TQL) and then stored in the Knowledge Base (KB) of the system. This is shown in Example 1, above the line. Later, when someone inputs a question, the question will also first be coded using TQL, but either the subject or one of the objects in the sentence may then be wildcards that should be matched against facts in the existing KB.

[1] http://www.ncbi.nlm.nih.gov/entrez/query.fcgi
[2] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/topics/Corpus/GTB.html

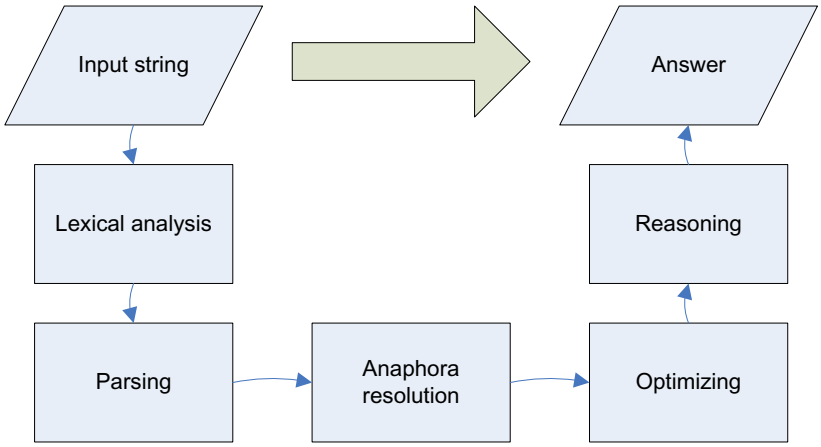

Fig. 1. Data flow in the TUC System

Example 1.

Statement : "CCK activates Gastrin."
Update to KB : activate(cck,gastrin).
Question : "What activates Gastrin?"
Answer : "CCK"

In this case it is very obvious that "What" is the placeholder for the answer, and also that it must be substituted with "CCK" to match the existing fact. So even if this is a very simple example, the method works in the same way also for more complex sentences. The only requirement is that the question is stated in a similar grammatical form as the factual statement.

3.1 GeneTUC Lexical Analysis

The lexical analysis in GeneTUC changes the input sentences from a long list of characters into tokens (words) and sentence delimiters. The current set of sentence delimiters includes the following:

Period	Colon	Semi Colon	Question Mark	Exclamation Mark
.	:	;	?	!

In the process of making the tokens, no distinction is made between upper and lower case characters, so the input to the syntactical analysis is a set of all lower case tokens.

3.2 GeneTUC Grammar and Syntactical Analysis

The GeneTUC grammar is what we call ConSensiCAL. That means it is a Context Sensitive Categorial Attribute Logic grammar formalism. It is based on the Prolog Definite Clause Grammar (DCG) with a few extensions to handle categorial movement and gaps etc. See [5] for more details on the Prolog programming language for Natural Language Processing (NLP).

Categorial Grammar. TUC is inspired by Categorial Grammar which allows *gaps* in the sentence. This mechanism is very easy to use when parsing sentences like in the following examples:

Example 2.

Input: What activates Gastrin?

Grammar for Question, using Statement:

Statement → NounPhrase VerbPhrase

Question → *what* Statement\NounPhrase

VerbPhrase → Verb NounPhrase

...

Example 3.

Input:
Results of preliminary studies, which we have conducted,
suggest that use of this agent is useful.

Grammar (Forward Application):

NounPhrase → Det Nominal RelativeClause

RelativeClause → RelativePron Statement/NounPhrase

RelativePronoun → *that*|*which*|*who*

...

Example 4.

Input: A gene signal that results in production of proteins occurs.

Grammar (Backward Application):

Statement → NounPhrase RelativeClause

RelativeClause → RelativePronoun Statement\NounPhrase

...

Example 5.

Input: A gene signal resulting in protein production occurs.

Grammar for Gerund:

RelativeClause → Verb-*ing* RelativeClause*that*Verb-*s*

...

Example 2 shows how the *What-Question* from Example 1 can be parsed using the existing grammar rules for Statement. It states that a *"what-question"* consists of the word "what" followed by a *Statement*, which is missing the leading *Noun Phrase (NP)*. This kind of Categorial *movement* makes it possible to connect the missing NP in the question ("what") with the actual NP in a corresponding fact statement ("CCK"), and then give a correct answer to the natural language query. This use of Backward (\) and Forward (/) application also reduces the number of grammar rules needed, since every new rule for statements implicitly creates corresponding new rules for questions.

In Example 3 the use of Forward application is shown. In GeneTUC, Forward application also includes Inward application, so "S/NP" also means that the NP can be missing anywhere in the Statement.

In Example 4, Backward application is used to define a Relative Clause. The missing NP in the Relative Clause can be found by going back to the NP that is preceding the Relative Clause.

Example 5 shows a different form of the sentence from Example 4. With the help of Backward application only one rule is needed to change this *gerund*

sentence into a RelativeClause that can then be parsed by the grammar given in Example 3. This rule is context sensitive, meaning that *ing*-verbs like "resulting" can only be substituted by "that verb-*s*" phrases, like "that results", when the parser is already expecting to see a RelativeClause.

3.3 Reducing the Parsing Time

In GeneTUC parsing time is reduced by the use of *cuts* in the Prolog code. This means that once a specific rule, for example *Noun Phrase (NP)*, has been successfully applied to a part of the input sentence, this part of the sentence is *committed* and can not be parsed again even if the following rules causes the parser to fail. Usually, failing on one possible parsing attempt would cause the parser to back-track and use the rule on a different span of words to produce a different and successful NP. This kind of backtracking can be very computationally expensive, especially with highly ambiguous input, so *cuts* greatly reduces the parse time. The cuts are placed manually in strategic places in the code, based on experience from previous parsing of *run-away* sentences. Usually, the cuts do not affect the final result from the parser, but some phenomena can cause the parser to fail because the assumed partial parsing result before the cut is incompatible with the rest of the sentence. One such phenomenon, which is hard to parse even for humans, is *garden path sentences* [13].

4 Methods

The main goal of this research was to evaluate the GeneTUC parser on the GENIA corpus. Since GeneTUC and GENIA were not made using any common grammar standard, a lot of modifications in GeneTUC were needed. These adaptations can be thought of as a (manual, not statistical) training process for GeneTUC, but in order to measure how the GeneTUC parser will perform on unseen data, different parts of the GENIA Tree Bank (GTB) was used for training and testing, i.e. we used 300 abstracts (GTB300) for training and the remaining 200 abstracts (GTB200) for testing.

The training phase of the project is described in the following subsections, and includes the following tasks:

- Dictionary building. Adding all terms from GENIA to the GeneTUC dictionary.
- Ontology building. Mapping from GENIA to GeneTUC dictionary classes.
- Adding other missing words manually, with the help of Bioogle.
- Input new verb templates, based on predicate argument structures seen in GENIA.

4.1 Updating GeneTUC Lexicon from GENIA

Since the goal is to test the parser, errors connected to the Tokenizer, POS tagger or other parts of the system should be removed. The ideal approach would be

to use the tokenized and POS-tagged version of GENIA as input to GeneTUC, but this was not feasible since GeneTUC is based on plain ASCII-text input. Also, it would take more manpower/time than available in this project to split the tight connection between tokenizing, tagging and parsing in TUC, just to test if it would be useful to do so later. Instead, the GENIA multi-word-terms were added to the GeneTUC dictionary, trying to guide it into using the same tokenization as in the GENIA gold file. This was only successful in around 20% of the sentences, so we reduced the test set to only include sentence that were similarly tokenized and tagged by GENIA and GeneTUC.

During the process of importing all the terms from GENIA into the TUC, several considerations had to be made:

1. *Plural Forms.* Plural words were changed into their singular (stem) form by simple rules like: remove the final s from all words. Exceptions to this simple rule had to be made for words like virus (already singular), viruses (remove -es) and bodies (change -ies to y).
2. *Proper Names or Common Nouns.* Another point is that plural forms should exist only as ako[3] relations (class concepts), and *not* as isa[4] relations (proper names).
3. *Duplicate Entries.* Changing plural forms into singular forms often leads to duplicate entries in the dictionary, since the singular form
4. *Short Ambiguous Terms.* The title sentence "Cloning of ARE-containing genes by AU-motif-directed display" causes a problem since TUC does not distinguish "ARE" and "are". Words like "are", "is", "a" and so on are therefore removed from the dictionary.

4.2 Updating the GeneTUC Semantic Network

As mentioned in the introduction, GeneTUC is a deep parser, requiring that all the input words are already in the dictionary. In order to compare just the parsing performance of GeneTUC with other systems, other error sources such as incomplete lexical tagging was reduced by importing all named entities from GENIA to GeneTUC. When new words are added to GeneTUC, it is also necessary to specify which semantic class they belong to, so a mapping between GENIA ontology and the ontology of GeneTUC was needed (Figure 2). One alternative way was to simply add all the ontology terms of GENIA (37) to GeneTUC, but many of the terms were already present in both systems, with slightly different classifications. We could also have changed the GeneTUC ontology terms to match those of GENIA, but that would have made many of the existing verb templates in GeneTUC useless or wrong, making this approach unappealing. The final choice was to create a mapping from GENIA ontology terms to existing GeneTUC ontology terms, as shown in Figure GENIA ontology. The GENIA term "other_name" and the corresponding GeneTUC term "stuff" are "bag" definitions, meaning that no effort was made to distinguish the terms that

[3] ako = A Kind Of (subclass of a class).
[4] isa = Is A (instance of a class).

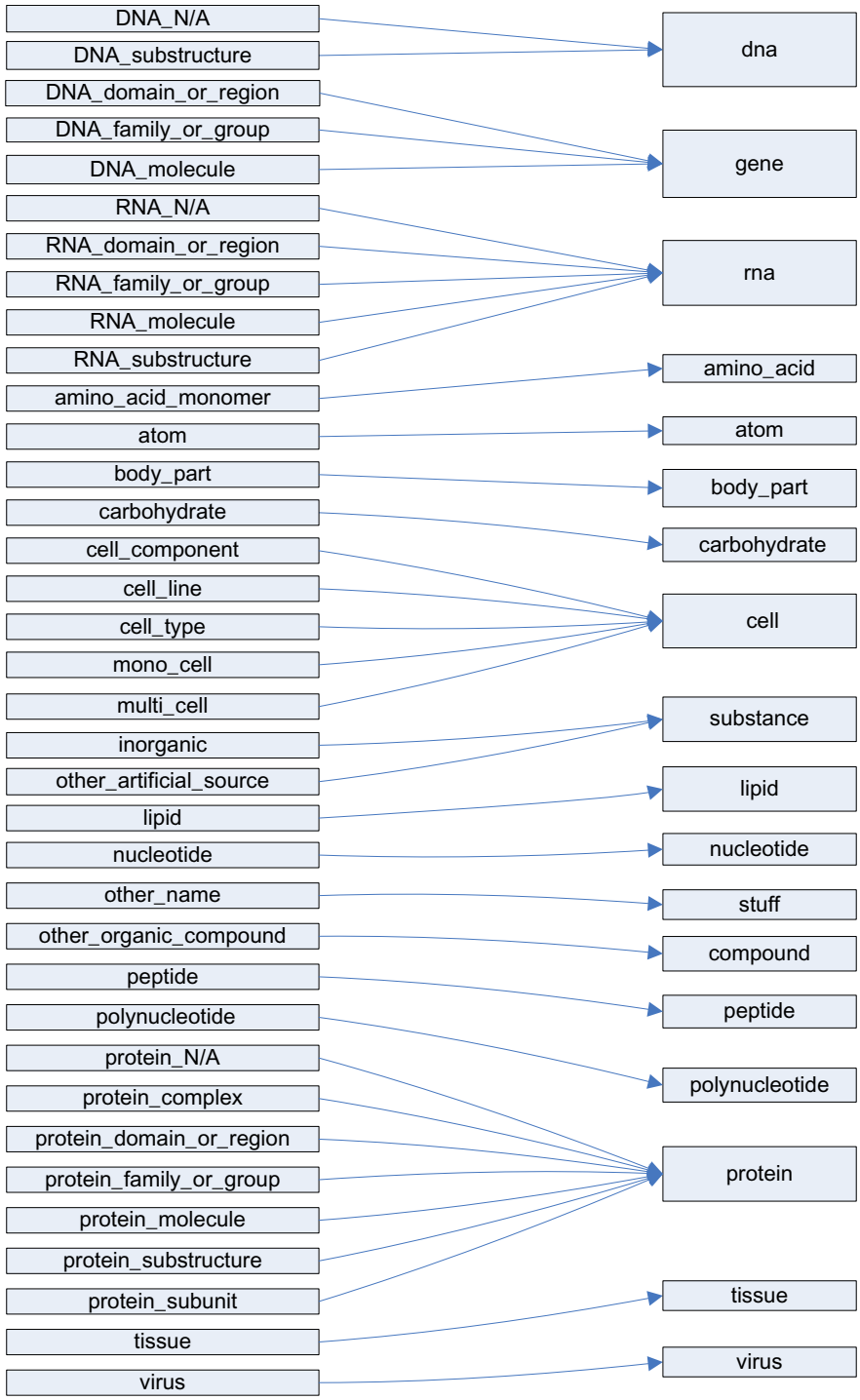

Fig. 2. Conversion from GENIA to GeneTUC Ontology

did not belong to one of the other classes. Many of these terms can easily be put into other existing GeneTUC classes, just by matching the last noun in the noun phrases as in the following example:

Example 6.
"nf_kappa_b_activation" *ako* activation
'0_95_kb_id_3_transcript' *ako* transcript
'17_amino_acid_epitope' *ako* epitope
asp_to_asn_substitution *ako* substitution

4.3 Adapting TUC to GTB/PTB Syntax Standard

Since we wanted to use the GENIA Tree Bank (GTB) for evaluating the GeneTUC parser, we needed to make sure that the output from the GeneTUC parser was in the same format as the parse trees from the GTB. This is a somewhat complicated process, since the TUC parser uses an internal syntax representation that is tightly connected to the semantics of the sentence, and this representation is different from the GTB syntax in a few important aspects. For example, the Categorial movement and gap mechanisms are implemented in TUC by doing parsing and reparsing. That means that the sentence will be parsed once, and then gaps will be filled with the syntax from the first parse, before the new resulting sentence is parsed again. This means that traces of the moved phrases will appear both where the phrase was originally, and where the gap was in the resulting parse tree. This leads to parse trees that look slightly different from the GTB parse trees, where each gap is given an Identifier (ID) and then the corresponding syntactical phrase is given the same ID-number. As long as no effort is made to implement this gap-ID system of the GTB grammar in TUC, these differences will lead to lower accuracy values in the evaluation, even though the parsing result is actually correct. To prevent this from happening, the internal workings of TUC had to be modified to produce output exactly equal to the expected output, and some pre- and post-processing scripts had to be made in order to smooth out the remaining systematical differences that could not be changed inside TUC. Still, some traces of these problems may be left in the final evaluation scores.

The creation of the grammar is currently done 100% manually. It is a slow and long-lasting job, but it ensures that all the rules are meaningful. The creation of a new rule is always rooted in the existence of old rules, as was shown in Examples 2 & 3.

4.4 EvalB and Tokenization

EvalB[5] was used for calculation of precision and recall scores for GeneTUC against the GENIA corpus. It requires that the number of tokens in the output text has to match the number of tokens in the input/gold text exactly. This is a challenge to GeneTUC, since it ignores the characters listed in Example 7.

Example 7. Ignored Characters: " : , & % { } ⟨ ⟩ [ ] (. . .)

[5] http://nlp.cs.nyu.edu/evalb/

Also, single tokens (like IL-2) are sometimes turned into two or three separate tokens ("il", "-" and "2"), because of hyphens. This happens when the word is not specifically defined in the dictionary as being just one word/token. Since the GTB is already tokenized and stored in XML format, the correct tokenization is known. The challenge is to ensure that TUC produces the expected output, even if the internal modules are using different tokens. Other features of GeneTUC that makes the comparison difficult is that some *noisewords* are removed from the text, and long Noun Phrases can sometimes be substituted with Canonical Identifiers.

There are two obvious solutions to this problem: The first is to use the tokenized version of GTB, instead of the plaintext version. The problem then, is that we have to circumvent the tokenization module (and lexical analysis) in TUC, and this might introduce problems in the later modules, for example because of ignored characters that were previously handled by the lexical module. Another example of problems introduced if the original tokenization is used, is parentheses with their contents. In the current implementation all level-1 parentheses are removed together with everything inside them, since this is usually ungrammatical constructions.

The second solution to the tokenization problem is to make a new plaintext version of the text, from the tokenized xml-version. In the new plain text version, all tokens containing hyphens and other troublesome characters, will be substituted by a new token using underscore (_) or some other character instead of hyphen, so that the lexical module does not split these token into extra tokens. In the opposite case, where the gold text contains more tokens than what TUC produces, we have to introduce some dummy tokens. These tokens can then act as placeholders for tokens that TUC ignores (and removes), like parentheses with all their content/tokens.

4.5 EvalB Comparing Syntax Trees

Using the tokens in the sentence as basis for scoring, EvalB performs a strict evaluation. Any case of tokenization different from the "golden" tokenization renders the parse incomparable; the sentences where the number of golden tokens and test tokens are unequal lead to an *error*. The same is true for those where the golden token and test token are character wise different. If the number of tokens equals zero (i.e. the sentence did not parse successfully), the sentence is *skipped*. Both *skip*- and *error*-sentences are ignored when calculating the score. The program provides a tolerance limit for how many incomparable sentences that are ignored.

Bracketing is measured from token[m] to token[n], where a *match* means those brackets covering the correct tokens, and having a correct *label*. The matches enable measurements of:

- Recall (the ratio between matched brackets and all brackets in the gold file)
- Precision (the ratio between matching brackets and all brackets in the test file)

- Crossing (the number of test-file-brackets exceeding the span of a matching bracket in the gold file, divided by the total number of brackets in test file)
- Tagging accuracy (the ratio of correct labeled tokens over the total number of tokens)

EvalB performs strict evaluation of the parse, as it originally was intended as a solemn bracketing evaluation program. Bracketing scores of GeneTUC may be reduced because of a right-orientation implied by the grammar of TUC (always preferring right-attachment unless it is semantically erroneous).

5 Results

This section shows the results from the training and test phases. Table 1 shows how much the performance of GeneTUC increased when the dictionary was extended with all the terms from GENIA. Table 2 shows that there was no significant difference in parsing scores between importing all the GENIA terms (36.692) or just the terms from GTB200 that were reported as unknown by GeneTUC

Table 1. Statistics parsing attempts before and after adding GENIA dictionary

Measurement	**Dictionary**	
Description	**Original**	**+GENIA**
Number of sentences:	2591	2591
Successful parses:	318	1531
Success rate:	12.3%	59.1%
Sources of Failure		
Dictionary:	1989	68
Grammar:	520	1126
Time out:	32	144
Processing time:	0.5 hrs	4.5 hrs

Table 2. Test results from EvalB

Measurement	**Dictionary**	
Description	**+GENIA**	**+GTB200**
Number of sentences	1759	1759
Error sentences	518	565
Skip sentences	1037	843
Valid sentences	204	351
Bracketing Recall	49.8%	53.9%
Bracketing Precision	69.0%	70.6%
Complete match	0.49%	1.14%
Average crossing	1.27	1.47
No crossing	47.1%	44.7%
2 or less crossing	79.9%	79.5%
Tagging accuracy	82.0%	86.0%

(8.175). In terms of input to EvalB, it was possible to compare almost twice as many sentences when only the GTB200 dictionary was used. This is mainly because GeneTUC was given fewer chances to rewrite complex multi-word tokens, and thereby creating better accordance between the output and the gold file.

6 Related Work

Other recent and related IE techniques for biomedical literature includes systems using dynamic programming [8] or supervised machine learning [22] to find protein-protein-relations in molecular biology texts. The machine learning approach uses both parse trees and dependent tree structures as features, but they report that simple lexical features contribute more to the promising F-measure of 90.9. Other systems use predicate-argument structure patterns [23] or new self made architecture [21] to do more general Information Extraction from this kind of free text sources.

6.1 GeneTUC, Bioogle and GProt

This paper showed how important a proper dictionary is to this kind of semantic parsers. A new way to automatically build dictionaries with ontology information is presented as Bioogle in [18]. Bioogle[6] is a simple system that uses Google to determine the semantic class of a word, for example "CCK is a protein", so that it can be added to the semantic hierarchy (or dictionary) in a correct way.

GProt [17], is another application that is built on top of the Google API, like Bioogle. GProt[7] provides a way of automatically extracting information from the (biomedical research) literature. Most of the literature is already indexed in MEDLINE, and therefore also by Google and other major search engines. See [16] for more details and a description of how to access the online versions of Bioogle and GProt.

7 Discussion

This section points out some of the lessons learned during the parsing project. This includes remarks about titles as a different sentence type and a discussion about the results presented in the two previous sections.

7.1 Sentence Types

MEDLINE (GTB) contains two fundamentally different sentence types: Titles and normal sentences. The titles are special, because they sometimes just state the object of the experiment, without the subject and verb phrase that should have started the sentence. Subject and verb-less sentences were already handled by GeneTUC before, but during this work we added a special "\title"-tag for

[6] http://furu.idi.ntnu.no:8080/bioogle/
[7] http://furu.idi.ntnu.no:8080/gprot/

the titles, so that we can implement some special processing of titles later. The first function we added to the "\title"-tag was resetting the temporary anaphoric database, so that terms like "the protein", "this" and "which" do not map to names or events in any previously parsed abstracts.

7.2 Comparing Different Systems

It turned out that evaluating the GeneTUC parser on a PTB gold standard file was harder than first expected. The main reason for this is that TUC was never meant to output PTB style tags in the first place. Also, there is not always a clear boundary between lexical, syntactical and semantic analysis. Of course, there are both advantages and disadvantages to this approach.

The problem that we encountered because of the tight connection between the modules in GeneTUC, is that it is very hard to construct output with the exact number of tokens as in the input text. TUC is based on receiving plain text input, and does its own tokenization and optimization of the text before passing it on to the parser. We could perhaps have used the already tokenized text as input, but this would introduce the parser to problems it is not meant to handle in the current configuration of the system. It would be much easier to cooperate with other researchers if the modules of GeneTUC were truly separate from each other, but it can also be argued that the good performance by a text processing system like this is really dependent on tight communication between the modules.

Tokenization is usually done before, and separate from, parsing, but sometimes it is necessary to do preliminary parsing in order to determine word and sentence boundaries. Parsing is usually done before semantic analysis, but sometimes it is important to know the semantic properties of a word in order to reduce the ambiguity, and thereby the parsing time. Maybe the time has come to start integrating the different modules more? This will require some effort to ensure that cooperation between different researchers is still possible, for example through the use of new standards/protocols for future parsers.

8 Conclusion

There is a great need for systems that can support biologists (and any other research) in dealing with the ever increasing information overload in their field. This project has proven that both the Google API and the GeneTUC systems are important pieces that can play a role in making the dream of real automatic Information Extraction come true in the not so distant future.

The precision and recall scores achieved by GeneTUC on general parsing are not very high compared to pure parsers like [4,10,7], but that does not mean that these systems are better than GeneTUC, because GeneTUC also performs deeper analysis such as anaphora resolution [2,14]. The other systems consist of Context-Free Grammar (CFG) parsers that give only phrase structures as output. There are also some systems that use CCG parsers [3] or HPSG parsers [11] that can give predicate argument structures in addition to phrase structures,

but they still do not perform anaphora resolution or question-answering, like GeneTUC does.

Acknowledgements

The first author would like to thank all the people who made the writing of this chapter possible. Especially, Professor Tsujii who invited me to his lab in Tokyo, and all his brilliant co-workers who helped me with anything related to the GENIA corpus. Much gratitude also goes to my co-supervisors Amund Tveit and Astrid Lægreid for continuous support, and for pointing out good opportunities.

References

1. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.
2. J. Castano, J. Zhang, and J. Pustejovsky. Anaphora resolution in biomedical literature. In *International Symposium on Reference Resolution*, 2002.
3. Stephen Clark, Julia Hockenmaier, and Mark Steedman. Building Deep Dependency Structures with a Wide-Coverage CCG Parser. In *Proceedings of ACL'02*, pages 327–334, 2002.
4. Andrew B. Clegg and Adrian J. Shepherd. Evaluating and integrating treebank parsers on a biomedical corpus. In *Proceedings of the ACL Workshop on Software 2005*, 2005.
5. Michael A. Covington. *Natural Language Processing for Prolog Programmers*. Prentice-Hall, Englewood Cliffs, New Jersey, 1994.
6. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.
7. Tadayoshi Hara, Yusuke Miyao, and Jun'ichi Tsujii. Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In *IJCNLP 2005: Second International Joint Conference on Natural Language Processing*, 2005.
8. Minlie Huang, Xiaoyan Zhu, Yu Hao, Donald G. Payan, Kunbin Qu, and Ming Li. Discovering patterns to extract protein-protein interactions from full texts. *Bioinformatics*, 20(18):3604–3612, Dec 12 2004.
9. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
10. Matthew Lease and Eugene Charniak. Parsing biomedical literature. In *Second International Joint Conference on Natural Language Processing (IJCNLP'05)*, 2005.
11. Yusuke Miyao and Jun'ichi Tsujii. Deep linguistic analysis for the accurate identification of predicate-argument relations. In *Proceedings of COLING 2004*, pages 1392–1397, 2004.
12. R. O'Donovan, M. Burkea, A. Cahill, J. van Genabith, and A. Way. Large-Scale Induction and Evaluation of Lexical Resources from the Penn-II Treebank. In *Proceedings of the 42nd Annual Meeting of the ACL.)*, pages 368–375, Barcelona, Spain, July 21-26 2004. Association for Computational Linguistics.

13. Lee Osterhout, Phillip J. Holcomb, and David A. Swinney. Brain potentials elicited by garden-path sentences: Evidence of the application of verb information during parsing. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 20(4):786–803, 1994.
14. J. Pustejovsky, J. Casta, J. Zhang, B. Cochran, and M. Kotecki. Robust relational parsing over biomedical literature: Extracting inhibit relations. In *Pacific Symposium on Biocomputing*, 2002.
15. Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU München, Germany, April 2003.
16. Rune Sætre, Martin T. Ranang, Tonje S. Steigedal, Kamilla Stunes, Kristine Misund, Liv Thommesen, and Astrid Lægreid. Webprot: Online Mining and Annotation of Biomedical Literature using Google. In Tuan D. Pham, Hong Yan, and Denis I. Crane, editors, *Advanced Computational Methods for Biocomputing and Bioimaging*. Nova Science Publishers, New York, USA, 2006.
17. Rune Sætre, Amund Tveit, Martin Thorsen Ranang, Tonje Strømmen Steigedal, Liv Thommesen, Kamilla Stunes, and Astrid Lægreid. GProt: Annotating Protein Interactions Using Google and Gene Ontology. In *Lecture Notes in Computer Science: Proceedings of the Knowledge Based Intelligent Information and Engineering Systems (KES2005)*, volume 3683, pages 1195 – 1203, Melbourne, Australia, August 2005. KES 2005, Springer.
18. Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal, and Astrid Lægreid. Semantic Annotation of Biomedical Literature using Google. In Dr. Osvaldo Gervasi, Dr. Marina Gavrilova, Dr. Youngsong Mun, Dr. David Taniar, Dr. Kenneth Tan, and Dr. Vipin Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO 2005)*, volume 3482 (Part III) of *Lecture Notes in Computer Science (LNCS)*, pages 327–337, Singapore, May 9-12 2005. Springer-Verlag Heidelberg.
19. Yuka Tateishi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. Syntax Annotation for the GENIA corpus. In *Proceedings of the IJCNLP 2005*, Korea, October 2005.
20. Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from . In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, pages 1161–1161, Tokyo, Japan, April 2005. IEEE Press (Electronic Publication).
21. Aditya Vailaya, Peter Bluvas, Robert Kincaid, Allan Kuchinsky, Michael Creech, and Annette Adler. An architecture for biological information extraction and representation. *Bioinformatics*, 21(4):430–438, 2005.
22. Juan Xiao, Jian Su, and GuoDong Zhouand ChewLim Tan. Protein-protein interaction extraction: A supervised learning approach. In *Semantic Mining in Biomedicine (SMBM)*, 2005.
23. Akane Yakushiji, Yusuke Miyao, Yuka Tateishi, and Junichi Tsujii. Biomedical information extraction with predicate-argument structure patterns. In *Semantic Mining in Biomedicine (SMBM)*, 2005.
24. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Part III

Synopsis

Chapter 4

Errata

Unfortunately, we are not perfect, so there are some errors in our publications. The once that have been located so far are listed below. The page numbers refer to the original page numbering, so page 328 in Paper III means page 328 as printed in the original version of Paper III, which starts on page 49 in this thesis.

4.1 Paper I

All occurrences of "microbiology" or "microbiological" should be substituted with the term "molecular biology".

4.2 Paper II

The calculated scores in table 3 for precision, recall and F-measure were wrong. A correct version of the table is given here:

Classifier	TP/TN	FP/FN	Prec/Rec/F	CA
N.Bayes	**6**/*120*	*67*/**7**	8/**46**/14	*63*
Majority	*0*/**187**	**0**/*13*	*NA*/*0*/*NA*	**94**
SVM Lin	*0*/**187**	**0**/*13*	*NA*/*0*/*NA*	**94**
SVM Pol	**6**/159	28/**7**	18/**46**/**26**	83
SVM rbf	3/174	13/10	**19**/23/21	89
SVM Sig	*0*/186	1/13	0/*0*/*NA*	93

4.3 Paper III

page 328: Table, example 3: occured → occurred

page 328: Research Hypothesis, reference [?] → [32].

page 329: seventh last line, seperate → separate

4.4 Paper IV

page 1202: Reference 16: Strœmmen → Strømmen

4.5 Paper V

page 93: jcb.org 32, Biological Chemistry → Cell Biology

4.6 Paper VI

page 71: Example 1 is a constructed example, and there is no biological evidence that "CCK activates Gastrin" in PubMed.

Chapter 5

Concluding Remarks

The final conclusions are given in Section 5.1 and an outline of some possible avenues for future work are given in Section 5.2. Section 5.3 gives a personal view on some important challenges facing the field of BioNLU in the near future.

5.1 Conclusions

Looking back at the main research question:

> *Can computer programs like GeneTUC understand text about molecular biology as well as humans can?*

The simple answer to the question is, at present, "no". Still, this thesis has shown that new search technology (like the Google API) can be used in the building of a system that understands at least large parts of the molecular biology texts, and at greater speed than humans can, too.

An NLP programmer with shallow knowledge in molecular biology needs a substantial amount of time to truly understand many of the sentences in this domain. Much time is spent looking at other sources than the text itself, and on building a conceptual model covering the entities that are described in the text. When this work is done, the programmer can "teach" the computer to parse similar sentences, just as fast as he can understand them himself. Still, even if the computer can *understand* more than the programmer in this case, the biologists are already much better than both the program and the programmer at understanding such sentences. So instead, the focus should be on making systems that can help the biologists structure the knowledge themselves, and

turning it into a computer readable form, for example by building ontologies. The first steps in this approach were explored in paper III and IV, and the implementation described in paper V.

To summarize, the key achievements and discoveries related to the research questions are:

RQ1 Biological entity names can be automatically extracted with the Bioogle system, and used in full parsing by GeneTUC.

RQ2 GeneTUC can now understand more than 50% of the sentences in the GENIA corpus, and answer simple questions from the logic code that is generated.

RQ3a Bioogle is currently the only known application to use Google API to aid programmers and biologists in the construction of dictionaries and ontologies for human protein names.

RQ3b GeneTUC is the only known system that uses natural language through the entire pipeline from parsing to question answering in this domain.

Other systems perform better on single subtasks in the GeneTUC system, like tagging, parsing and information extraction, but they don't offer the capabilities that GeneTUC has in addition, like a natural language interface and anaphora resolution.

5.2 Future Work

During the work herein, several opportunities for possibly fruitful extensions have been encountered.

5.2.1 Semantics

In an ideal world, it should be easy to add all the necessary terms to the semantic part of the system, but this is not the case. The main reason for this is that the biologists themselves have not done enough standardization work yet, especially when it comes to gene-ontologies and protein interaction naming. This means that every time something is added to the semantic network of GeneTUC, there is a chance that it will have to be changed later, because our view of the world is still changing. This is the same problem that for example the Gene Ontology consortium faces. Some early "guesses" turn out to be erroneous, and if they are not corrected soon, they become a credibility problem.

When moving from a language that is highly ambiguous, like English, to the language of logic, with only one acceptable interpretation, there must be some consensus about what the actual meaning of the given natural language constructs are. This consensus is not yet established in the molecular biology domain, but international work is in progress. Also for the GeneTUC system, it is important to follow these widely accepted guidelines, as they emerge.

5.2.2 Syntax

The same thing can be said in many ways. TUC currently requires questions to be stated in the same basic form or syntax as the corresponding input fact was, in order to be able to find the answer. This is an argument for using entire articles instead of just abstracts as input, since the same important facts will usually be stated many times, in different forms, throughout an entire article. That should make it easier to answer questions, since there will be more than one accepted way to formulate the *right* question.

GeneTUC is currently more or less punctuation-free. It might be worth taking at least some punctuation into the TUC language. Especially *commas* would have the potential to solve a lot of ambiguity problems. A good argument against dealing with punctuation though, is that it will also be a new source of problems, not only solutions. The way people use punctuation is as ambiguous as the way they use the language in general. Still, it would probably be worth the time looking into what positive improvements punctuation treatment would bring with it.

In GeneTUC, all input is converted to lower case, to make the system case-insensitive, and more tolerant to individual authors different ways of using upper case in their publications and questions. However, in [5], Cohen shows that the F-measure improved by 15 points, without hurting recall more than 2 percent-points, when they added case sensitivity to the 300 most common English words, so this should be considered in future versions of GeneTUC as well.

5.3 Outlook

The field of NLP is almost five decades old now, but the field of BioNLU is still just starting to "take off". Much standardization and cooperation is needed before we will eventually see good and useful applications for the end users, namely the biologists. And on the way to making a working GeneTUC or similar systems, the methods from Bioogle will be useful additions to the more general NLP techniques.

Bibliography

[1] Tore Amble. Domain Modelling and Automated Natural Language Interfaces. In S.L Hansen, editor, *Topics in Knowledge Based NLP Systems, Workshop on KBS*, Copenhagen, Denmark, 1994. Copenhagen Business School, Samfundslitteratur.

[2] Tore Amble. BusTUC - a natural language bus route oracle. In *Proceedings of the 6th Applied Natural Language Processing Conference*, pages 1–6, Seattle, Washington, USA, April 29–May 4 2000. Association for Computational Linguistics.

[3] Anders Andenæs. GeneTUC, An NLP System for biomedical Texts. (Master Project) Norwegian University of Science and Technology, Norway, April 2000.

[4] Anders Andenæs. GeneTUC /j&ˀne-tük/. Master's thesis, Norwegian University of Science and Technology, Norway, December 2000.

[5] Aaron M. Cohen. Unsupervised gene/protein named entity normalization using automatically extracted dictionaries. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, pages 17–24, Detroit, USA, June 2005. Association for Computational Linguistics.

[6] The International Human Genome Mapping Consortium. A physical map of the human genome. *Nature*, 409:934–941, February 2001.

[7] J. Craig Venter et al. The sequence of the human genome. *Science*, 291(5507):1304–1351, 2001.

[8] M. A. Harris et al. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Research*, 32:D258–D261, January 2004.

[9] Bill Hibbard. Götterdämmerung. Dawn of the Human-made Gods. Draft: http://www.ssec.wisc.edu/ billh/g/chapter2.html, April 7th 2001.

[10] Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.

[11] Jung-jae Kim and Jong C. Park. BioIE: Retargetable Information Extraction and Ontological Annotation of Biological Interactions from the Literature. *Journal of Bioinformatics and computational Biology*, 2(3):551–568, 2004.

[12] Daniel Saul Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice Hall Series in Artificial Intelligence. Prentice Hall, Upper Saddle River, New Jersey 07458, USA, 2000. ISBN 0-13-095069-6.

[13] K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia USA, July 2002. IBM.

[14] Peter H. Raven, George B. Johnson, Jonathan B. Losos, and Susan R. Singer. *Biology*. McGraw-Hill, 1221 Avenue of the Americas, New York, NY 10020, USA, 7th ed. edition, 2005. ISBN 0-07-243731-6.

[15] Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.

[16] Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU München, Germany, April 2003.

[17] Alan Turing. Computing, machinery, and intelligence. *MIND Journal of the Mind Association*, LIX(236):433–460, October 1950.

[18] W. A. Woods. *Linguistic Structures Processing*, chapter Lunar rocks in natural English: Explorations in natural language question answering, pages 521–569. A. Zampolli (Ed.), North Holland, Amsterdam., 1977.

[19] Akane Yakushiji, Yuka Tateisi, Yusuke Miyao, and Jun'ichi Tsujii. Event extraction from biomedical papers using a full parser. In *Proceedings of Pacific Symposium on Biocomputing 2001*, pages 408–419, Hawaii, USA, January 2001.

[20] Zhiping Zheng. Question answering using web news as knowledge base. In *Proceedings International WWW Conference(11)*, Honolulu, Hawaii, USA, 2002.

Bibliography of Paper I

[1] Bennett, H. A., He, Q., Powell, K., and Schatz, B. R. (1999) **Extracting noun phrases for all of Medline**. In *AMIA '99 (American Medical Informatics Assoc) Conf*, Washington, DC.

[2] Brill, E. (1992) **A simple rule-based part-of-speech tagger**. In *Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing*, Trento, Italy pages 152-155

[3] Cohen, K. B., Dolbey, A. E., Acquaah-Mensah, G. K. and Hunter, L. (2002) **Contrast and variability in gene names**. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain*. Pages 14-20

[4] Collier, N., Park, H., Ogata, N., Tateishi, Y., Nobata, C., Ohta, T., Sekimizu, T., Imai, H., Ibushi, K. and Tsujii, J. (1999) **The GENIA project: corpus-based knowledge acquisition and information extraction from genome research papers**, in *Proc. of the Annual Meeting of the European Association for Computational Linguistics (EACL-99)*, pp.271-272, Norway

[5] Eriksson, G., Franzén K., Olsson, F., Asker, L., Lidén, P. (2002). **Using Heuristics, Syntax and a Local Dynamic Dictionary for Protein Name Tagging**. Human Language Technology Conference 2002, San Diego, USA. URL: http://www.sics.se/~fredriko/papers/hlt02_abstract.pdf

[6] Ohta, Y., Yamamoto, Y., Okazaki, T., Uchiyama, I., and Takagi, T. (1997) **Automatic Construction of Knowledge Base from Biological Papers**. In *Proc. of the Fifth International Conference on Intelligent Systems for Molecular Biology (ISMB'97)*, pages 218-225.

[7] Fukuda, K., Tsunoda, T., Tamura, A., and Takagi, T. (1998) **toward information extraction: Identifying protein names from biological papers**. In *Proc. of the Pacific Symposium on Biocomputing '98 (PSB'98), Hawaii*. Human Genome Center, Institute of Medical Science, University of Tokyo. Email: ichiro@ims.u-tokyo.ac.jp

[8] Hishiki, T., Collier, N., Nobata, C., Ohta, T., Ogata, N., Sekimizu, T., Steiner, R., Park, H. S., and Tsujii, J. (1998) **Developing NLP tools for genome informatics: An information extraction perspective**. In *Proc. of Genome Informatics*, pages 81-90, Tokyo, Japan. Universal Academy Press, Inc., Tokyo, Japan.

[9] Gross, Maurice. (1997) **the Construction of Local Grammars**. In *Finite-State Language Processing*, E. Roche & Y. Schabès (eds.), Language, Speech, and Communication, Cambridge, Mass.: MIT Press, pages 329-354

[10] Jenssen, T. K., Lægreid, A., Komorowski, J., and Hovig, E. (2001) **a literature network of human genes for high-throughput analysis of gene expression**. In *Nature Genetics*, 28(1):21-28

[11] Park, Jong C. (2001) **Using Combinatory Categorial Grammar to Extract Biomedical Information**. In *IEEE Intelligent Systems*, 6. Korea

Advanced Institute of Science and Technology. URL: http://computer.org/intelligent/ex2001/x6toc.htm, pages 62-67

[12] Proux, D., Rechenmann, F., Julliard, L., Pillet, V., and Jacq, B. (1998) **Detecting gene symbols and names in biological texts: A first step toward pertinent information extraction.** In Miyano, S. and Takagi, T., editors, *Ninth Workshop on Genome Informatics*, volume 9, pages 72-80, Tokyo, Japan.

[13] Pustejovsky, J., Castaño, J., Zhang, J., Kotecki, M. and Cochran, B. (2002) **Robust Relational Parsing over Biomedical Literature: Extracting Inhibit Relations.** In *Proceedings of the Pacific Symposium on Biocomputing*, pages 362-373. Hawaii, USA

[14] Sætre, Rune. (2002) **GeneTUC**. Technical report, Department of Computer and Information Science, Norwegian University of Science and Technology, 2002. URL: http://www.idi.ntnu.no/~satre/genetuc/GeneTUC.pdf

[15] Yakushiji, A., Tateisi, Y., Miyao, Y. and Tsujii, J. (2001) **Event Extraction from Biomedical Papers Using a Full Parser**. In *Proceedings of Pacific Symposium on Biocomputing 2001*, pages 408-419, Hawaii, USA

[16] CiteSeer. URL: http://citeseer.nj.nec.com/

[17] Gene Ontology. URL: http://www.geneontology.org/

[18] Google. URL: www.google.com

[19] GENATLAS. URL: www.dsi.univ-paris5.fr/genatlas/

[20] The Genome Database. URL: http://gdbwww.gdb.org/

[21] The Human Genome Organisation (HUGO) Gene Nomenclature Committee. URL: http://www.gene.ucl.ac.uk/nomenclature/

[22] LocusLink. URL: www.ncbi.nlm.nih.gov/LocusLink/

[23] Medstract Project. URL: www.medstract.org

[24] Pacific Symposium on Biocomputing. URL: http://psb.stanford.edu/psb02/

[25] PubMed Medline. URL: www.pubmed.gov

[26] Unitex. URL: http://www-igm.univ-mlv.fr/~unitex/

Bibliography of Paper II

References

1. Steffen Bickel, Ulf Brefeld, Lukas Faulstich, Jørg Hakenberg, Ulf Leser, Condrad Plake, and Tobias Scheffer. A Support Vector Machine classifier for gene name recognition. In *Proceedings of the EMBO Workshop: A Critical Assessment of Text Mining Methods in Molecular Biology*, March 2004.
2. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.
3. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Raymond J. Mooney, Yuk Wah Wong, Edward M. Marcotte, and Arun Kumar Ramani. Learning to Extract Proteins and their Interactions from Medline Abstracts. In *Proceedings of the ICML-2003 Workshop on Machine Learning in Bioinformatics*, pages 46–53, August 2003.
4. Razvan Bunescu, Ruifang Ge, Raymond J. Mooney, Edward Marcotte, and Arun Kumar Ramani. Extracting Gene and Protein Names from Biomedical Abstracts. Unpublished Technical Note, Machine Learning Research Group, University of Texas at Austin, USA, March 2002.
5. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.
6. J. Demsar and B. Zupan. Orange: From Experimental Machine Learning to Interactive Data Mining. White Paper, Faculty of Computer and Information Science, University of Ljubljana, 2004.
7. Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley-Interscience, 2nd edition, 2001.
8. Filip Ginter, Jorma Boberg, Jouni Jarvinen, and Tapio Salakoski. New Techniques for Disambiguation in Natural Language and Their Application to Biological Texts. *Journal of Machine Learning Research*, 5:605–621, June 2004.
9. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
10. Sittichai Jiampojamarn. Biological term extraction using classification methods. Presentation at Dalhousie Natural Language Processing Meeting, June 2004.
11. Thorsten Joachims. *Advances in Kernel Methods: Support Vector Learning*, chapter 11 - Making Large-scale SVM Learning Practical. MIT Press, 1999.
12. Sougata Mukherjea, L. Venkata Subramaniam, Gaurav Chanda, Sriram Sankararaman, Ravi Kothari, Vishal Batra, Deo Bhardwaj, and Biplav Srivastava. Enhancing a biomedical information extraction system with dictionary mining and context disambiguation. *IBM Journal of Research and Development*, 48(5/6):693–701, September/November 2004.
13. M. Narayanaswamy, KE Ravikumar, and K Vijay-Shanker. A biological named entity recognizer. In *Proceedings of the Pacific Symposium on Biocomputing 2003*, pages 427–438, 2003.
14. Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.

15. Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU München, Germany, April 2003.
16. Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal, and Astrid Lægreid. Semantic Annotation of Biomedical Literature using Google. In Dr. Osvaldo Gervasi, Dr. Marina Gavrilova, Dr. Youngsong Mun, Dr. David Taniar, Dr. Kenneth Tan, and Dr. Vipin Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO 2005)*, volume 3482 (Part III) of *Lecture Notes in Computer Science (LNCS)*, pages 327–337, Singapore, May 9-12 2005. Springer-Verlag Heidelberg.
17. Hagit Shatkay and Ronen Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.
18. Manabu Torii and K. Vijay-Shanker. Using Unlabeled MEDLINE Abstracts for Biological Named Entity Classification. In *Proceedings of the 13th Conference on Genome Informatics*, pages 567–568, 2002.
19. Amund Tveit, Magnus Lie Hetland, and Håvard Engum. Incremental and Decremental Proximal Support Vector Classification using Decay Coefficients. In *Proceedings of the 5th International Conference on Data Warehousing and Knowledge Discovery (DAWAK'2003)*, volume 2737 of *Lecture Notes in Computer Science (LNCS)*, pages 422–429. Springer-Verlag, 2003.
20. Limsoon Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.
21. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Bibliography of Paper III

References

1. Steffen Bickel, Ulf Brefeld, Lukas Faulstich, Jørg Hakenberg, Ulf Leser, Condrad Plake, and Tobias Scheffer. A Support Vector Machine classifier for gene name recognition. In *Proceedings of the EMBO Workshop: A Critical Assessment of Text Mining Methods in Molecular Biology*, March 2004.
2. C. Blaschke, MA. Andrade, C. Ouzounis, and A. Valencia. Automatic Extraction of biological information from scientific text: Protein-protein interactions. In *Proceedings of International Conference on Intelligent Systems for Molecular Biology*, pages 60–67. AAAI, 1999.
3. B. Boeckmann, A. Bairoch, R. Apweiler, MC. Blatter, A. Estreicher, E. Gasteiger, MJ Martin, K Michoud, C. O'Donovan, I. Phan, S. Pilbout, and M. Schneider. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic Acids Research*, 31(1):365–370, January 2003.
4. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.
5. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Raymond J. Mooney, Yuk Wah Wong, Edward M. Marcotte, and Arun Kumar Ramani. Learning to Extract Proteins and their Interactions from Medline Abstracts. In *Proceedings of the ICML-2003 Workshop on Machine Learning in Bioinformatics*, pages 46–53, August 2003.
6. Razvan Bunescu, Ruifang Ge, Raymond J. Mooney, Edward Marcotte, and Arun Kumar Ramani. Extracting Gene and Protein Names from Biomedical Abstracts. Unpublished Technical Note, Machine Learning Research Group, University of Texas at Austin, USA, March 2002.
7. Philipp Cimiano and Steffen Staab. Learning by Googling. *SIGKDD Explorations Newsletter*, 6(2):24–34, December 2004.
8. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.
9. Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, pages 178–186. ACM, 2003.
10. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, and Beatrice Alex. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. In *Proceedings of the BioCreative Workshop*, March 2004.
11. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. Submitted to BMC Bioinformatics, 2004.
12. Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. Submitted to Artificial Intelligence, 2004.

13. K. Fukuda, A. Tamura, T. Tsunoda, and T. Takagi. Toward information extraction: identifying protein names from biological papers. In *Proceedings of Pacific Symposium on Biocomputing*, pages 707–718, 1998.
14. Filip Ginter, Jorma Boberg, Jouni Jarvinen, and Tapio Salakoski. New Techniques for Disambiguation in Natural Language and Their Application to Biological Texts. *Journal of Machine Learning Research*, 5:605–621, June 2004.
15. Jun ichi Tsujii and Limsoon Wong. Natural Language Processing and Information Extraction in Biology. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 372–373, 2001.
16. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
17. Sittichai Jiampojamarn. Biological term extraction using classification methods. Presentation at Dalhousie Natural Language Processing Meeting, June 2004.
18. Vinay Kakade and Madhura Sharangpani. Improving the Precision of Web Search for Medical Domain using Automatic Query Expansion. Online, 2004.
19. Udo Kruschwitz. Automatically Acquired Domain Knowledge for ad hoc Search: Evaluation Results. In *Proceedings of the 2003 Intl. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*. IEEE, 2003.
20. Sougata Mukherjea, L. Venkata Subramaniam, Gaurav Chanda, Sriram Sankararaman, Ravi Kothari, Vishal Batra, Deo Bhardwaj, and Biplav Srivastava. Enhancing a biomedical information extraction system with dictionary mining and context disambiguation. *IBM Journal of Research and Development*, 48(5/6):693–701, September/November 2004.
21. M. Narayanaswamy, KE Ravikumar, and K Vijay-Shanker. A biological named entity recognizer. In *Proceedings of the Pacific Symposium on Biocomputing 2003*, pages 427–438, 2003.
22. David Parry. A fuzzy ontology for medical document retrieval. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32*, pages 121–126. ACM Press, 2004.
23. J. U. Pontius, L. Wagner, and G. D. Schuler. UniGene: a unified view of the transcriptome. *The NCBI Handbook*, 2003.
24. KD Pruitt and DR Maglott. RefSeq and LocusLink: NCBI gene-centered resources. *Nucleic Acids Research*, 29(1):137–140, January 2001.
25. Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.
26. Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU München, Germany, April 2003.
27. Urvi Shah, Tim Finin, and Anupam Joshi. Information Retrieval on the Semantic Web. In *Proceedings of CIKM 2002*, pages 461–468. ACM Press, 2002.
28. Hagit Shatkay and Ronen Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.
29. Lorraine Tanabe and W. John Wilbur. Tagging gene and protein names in biomedical text. *Bioinformatics*, 18(8):1124–1132, 2002.
30. Manabu Torii and K. Vijay-Shanker. Using Unlabeled MEDLINE Abstracts for Biological Named Entity Classification. In *Proceedings of the 13th Conference on Genome Informatics*, pages 567–568, 2002.

31. Yoshimasa Tsuruoka and Jun'ichi Tsujii. Probabilistic Term Variant Generator for Biomedical Terms. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 167–173. ACM, July/August 2003.
32. Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from . In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, pages 1161–1161, Tokyo, Japan, April 2005. IEEE Press (Electronic Publication).
33. Limsoon Wong. A Protein Interaction Extraction System. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 520–530, 2001.
34. Limsoon Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.
35. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.
36. Hong Yu, Vasileios Hatzivassiloglou, Andrey Rzhetsky, and W. John Wilbur. Automatically identifying gene/protein terms in MEDLINE abstracts. *Journal of Biomedical Informatics*, 35(5/6):322–330, October 2002.

Bibliography of Paper IV

References

1. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.
2. Philipp Cimiano and Steffen Staab. Learning by Googling. *SIGKDD Explorations Newsletter*, 6(2):24–34, December 2004.
3. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.
4. Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, pages 178–186. ACM, 2003.
5. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, and Beatrice Alex. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. In *Proceedings of the BioCreative Workshop*, March 2004.
6. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. Submitted to BMC Bioinformatics, 2004.
7. Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. Submitted to Artificial Intelligence, 2004.
8. Jun ichi Tsujii and Limsoon Wong. Natural Language Processing and Information Extraction in Biology. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 372–373, 2001.
9. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
10. Vinay Kakade and Madhura Sharangpani. Improving the Precision of Web Search for Medical Domain using Automatic Query Expansion. Online, 2004.
11. Udo Kruschwitz. Automatically Acquired Domain Knowledge for ad hoc Search: Evaluation Results. In *Proceedings of the 2003 Intl. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*. IEEE, 2003.
12. Eric P. G. Martin, Eric G. Bremer, Marie-Claude Guerin, Catherine DeSesa, and Olivier Jouve. Analysis of protein/protein interactions through biomedical literature: Text mining of abstracts vs. text mining of full text articles. In *Knowledge Exploration in Life Science Informatics (KELSI2004) Symposium*, number 3303 in Lecture Notes in Artificial Intelligence (LNAI), pages 96–108. Springer Verlag, 2004.
13. David Parry. A fuzzy ontology for medical document retrieval. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32*, pages 121–126. ACM Press, 2004.

14. Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.
15. Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU München, Germany, April 2003.
16. Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal, and Astrid Lægreid. Semantic Annotation of Biomedical Literature using Google. In Dr. Osvaldo Gervasi, Dr. Marina Gavrilova, Dr. Youngsong Mun, Dr. David Taniar, Dr. Kenneth Tan, and Dr. Vipin Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO 2005)*, volume 3482 (Part III) of *Lecture Notes in Computer Science (LNCS)*, pages 327–337, Singapore, May 9-12 2005. Springer-Verlag Heidelberg.
17. Urvi Shah, Tim Finin, and Anupam Joshi. Information Retrieval on the Semantic Web. In *Proceedings of CIKM 2002*, pages 461–468. ACM Press, 2002.
18. Hagit Shatkay and Ronen Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.
19. Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from . In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, pages 1161–1161, Tokyo, Japan, April 2005. IEEE Press (Electronic Publication).
20. Limsoon Wong. A Protein Interaction Extraction System. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 520–530, 2001.
21. Limsoon Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.
22. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Bibliography of Paper V

References

1. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.
2. Philipp Cimiano and Steffen Staab. Learning by Googling. *SIGKDD Explorations Newsletter*, 6(2):24–34, December 2004.
3. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.
4. Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, pages 178–186. ACM, 2003.
5. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, and Beatrice Alex. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. In *Proceedings of the BioCreative Workshop*, March 2004.
6. Shipra Dingare, Jenny Finkel, Christopher Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. Exploring the Boundaries: Gene and Protein Identification in Biomedical Text. Submitted to BMC Bioinformatics, 2004.
7. Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. Submitted to Artificial Intelligence, 2004.
8. Robert Hoffmann and Alfonso Valencia. A gene network for navigating the literature. *Nature Genetics*, 36(7):664, May 2004.
9. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
10. Vinay Kakade and Madhura Sharangpani. Improving the Precision of Web Search for Medical Domain using Automatic Query Expansion. Online, 2004.
11. Udo Kruschwitz. Automatically Acquired Domain Knowledge for ad hoc Search: Evaluation Results. In *Proceedings of the 2003 Intl. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*. IEEE, 2003.
12. Eric P. G. Martin, Eric G. Bremer, Marie-Claude Guerin, Catherine DeSesa, and Olivier Jouve. Analysis of protein/protein interactions through biomedical literature: Text mining of abstracts vs. text mining of full text articles. In *Knowledge Exploration in Life Science Informatics (KELSI2004) Symposium*, number 3303 in Lecture Notes in Artificial Intelligence (LNAI), pages 96–108. Springer Verlag, 2004.
13. David Parry. A fuzzy ontology for medical document retrieval. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32*, pages 121–126. ACM Press, 2004.
14. Rune Sætre. GeneTUC, A Biolinguistic Project. (Master Project) Norwegian University of Science and Technology, Norway, June 2002.

15. Rune Sætre, Amund Tveit, Martin Thorsen Ranang, Tonje Strømmen Steigedal, Liv Thommesen, Kamilla Stunes, and Astrid Lægreid. GProt: Annotating Protein Interactions Using Google and Gene Ontology. In *Lecture Notes in Computer Science: Proceedings of the Knowledge Based Intelligent Information and Engineering Systems (KES2005)*, volume 3683, pages 1195 – 1203, Melbourne, Australia, August 2005. KES 2005, Springer.
16. Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal, and Astrid Lægreid. Semantic Annotation of Biomedical Literature using Google. In Dr. Osvaldo Gervasi, Dr. Marina Gavrilova, Dr. Youngsong Mun, Dr. David Taniar, Dr. Kenneth Tan, and Dr. Vipin Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO 2005)*, volume 3482 (Part III) of *Lecture Notes in Computer Science (LNCS)*, pages 327–337, Singapore, May 9-12 2005. Springer-Verlag Heidelberg.
17. Urvi Shah, Tim Finin, and Anupam Joshi. Information Retrieval on the Semantic Web. In *Proceedings of CIKM 2002*, pages 461–468. ACM Press, 2002.
18. Hagit Shatkay and Ronen Feldman. Mining the Biomedical Literature in the Genomic Era: An Overview. *Journal of Computational Biology*, 10(6):821–855, 2003.
19. Jun'ichi Tsujii and Limsoon Wong. Natural Language Processing and Information Extraction in Biology. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 372–373, 2001.
20. Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from . In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, pages 1161–1161, Tokyo, Japan, April 2005. IEEE Press (Electronic Publication).
21. Limsoon Wong. A Protein Interaction Extraction System. In *Proceedings of the Pacific Symposium on Biocomputing 2001*, pages 520–530, 2001.
22. Limsoon Wong. Gaps in Text-based Knowledge Discovery for Biology. *Drug Discovery Today*, 7(17):897–898, September 2002.
23. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Bibliography of Paper VI

References

1. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Journal Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*, 2004.
2. J. Castano, J. Zhang, and J. Pustejovsky. Anaphora resolution in biomedical literature. In *International Symposium on Reference Resolution*, 2002.
3. Stephen Clark, Julia Hockenmaier, and Mark Steedman. Building Deep Dependency Structures with a Wide-Coverage CCG Parser. In *Proceedings of ACL'02*, pages 327–334, 2002.
4. Andrew B. Clegg and Adrian J. Shepherd. Evaluating and integrating treebank parsers on a biomedical corpus. In *Proceedings of the ACL Workshop on Software 2005*, 2005.
5. Michael A. Covington. *Natural Language Processing for Prolog Programmers*. Prentice-Hall, Englewood Cliffs, New Jersey, 1994.
6. J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, January 1996.
7. Tadayoshi Hara, Yusuke Miyao, and Jun'ichi Tsujii. Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In *IJCNLP 2005: Second International Joint Conference on Natural Language Processing*, 2005.
8. Minlie Huang, Xiaoyan Zhu, Yu Hao, Donald G. Payan, Kunbin Qu, and Ming Li. Discovering patterns to extract protein-protein interactions from full texts. *Bioinformatics*, 20(18):3604–3612, Dec 12 2004.
9. Tor-Kristian Jenssen, Astrid Lægreid, Jan Komorowski, and Eivind Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1):21–28, May 2001.
10. Matthew Lease and Eugene Charniak. Parsing biomedical literature. In *Second International Joint Conference on Natural Language Processing (IJCNLP'05)*, 2005.
11. Yusuke Miyao and Jun'ichi Tsujii. Deep linguistic analysis for the accurate identification of predicate-argument relations. In *Proceedings of COLING 2004*, pages 1392–1397, 2004.
12. R. O'Donovan, M. Burkea, A. Cahill, J. van Genabith, and A. Way. Large-Scale Induction and Evaluation of Lexical Resources from the Penn-II Treebank. In *Proceedings of the 42nd Annual Meeting of the ACL.)*, pages 368–375, Barcelona, Spain, July 21-26 2004. Association for Computational Linguistics.
13. Lee Osterhout, Phillip J. Holcomb, and David A. Swinney. Brain potentials elicited by garden-path sentences: Evidence of the application of verb information during parsing. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 20(4):786–803, 1994.
14. Tuan D. Pham, Hong Yan, and Denis I. Crane. *Advanced Computational Methods for Biocomputing and Bioimaging*, chapter WebProt: Online Mining and Annotation of Biomedical Literature using Google. Nova Science Publishers, New York, USA, 2006.
15. J. Pustejovsky, J. Casta, J. Zhang, B. Cochran, and M. Kotecki. Robust relational parsing over biomedical literature: Extracting inhibit relations. In *Pacific Symposium on Biocomputing*, 2002.

16. Rune Sætre. Natural Language Processing of Gene Information. Master's thesis, Norwegian University of Science and Technology, Norway and CIS/LMU München, Germany, April 2003.
17. Rune Sætre, Amund Tveit, Martin Thorsen Ranang, Tonje Strømmen Steigedal, Liv Thommesen, Kamilla Stunes, and Astrid Lægreid. GProt: Annotating Protein Interactions Using Google and Gene Ontology. In *Lecture Notes in Computer Science: Proceedings of the Knowledge Based Intelligent Information and Engineering Systems (KES2005)*, volume 3683, pages 1195 – 1203, Melbourne, Australia, August 2005. KES 2005, Springer.
18. Rune Sætre, Amund Tveit, Tonje Strømmen Steigedal, and Astrid Lægreid. Semantic Annotation of Biomedical Literature using Google. In Dr. Osvaldo Gervasi, Dr. Marina Gavrilova, Dr. Youngsong Mun, Dr. David Taniar, Dr. Kenneth Tan, and Dr. Vipin Kumar, editors, *Proceedings of the International Workshop on Data Mining and Bioinformatics (DMBIO 2005)*, volume 3482 (Part III) of *Lecture Notes in Computer Science (LNCS)*, pages 327–337, Singapore, May 9-12 2005. Springer-Verlag Heidelberg.
19. Yuka Tateishi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. Syntax Annotation for the GENIA corpus. In *Proceedings of the IJCNLP 2005*, Korea, October 2005.
20. Amund Tveit, Rune Sætre, Tonje S. Steigedal, and Astrid Lægreid. ProtChew: Automatic Extraction of Protein Names from . In *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE 2005, in conjunction with ICDE 2005)*, pages 1161–1161, Tokyo, Japan, April 2005. IEEE Press (Electronic Publication).
21. Aditya Vailaya, Peter Bluvas, Robert Kincaid, Allan Kuchinsky, Michael Creech, and Annette Adler. An architecture for biological information extraction and representation. *Bioinformatics*, 21(4):430–438, 2005.
22. Juan Xiao, Jian Su, and GuoDong Zhouand ChewLim Tan. Protein-protein interaction extraction: A supervised learning approach. In *Semantic Mining in Biomedicine (SMBM)*, 2005.
23. Akane Yakushiji, Yusuke Miyao, Yuka Tateishi, and Junichi Tsujii. Biomedical information extraction with predicate-argument structure patterns. In *Semantic Mining in Biomedicine (SMBM)*, 2005.
24. Hong Yu, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W. John Wilbur. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In *Proceedings of the AMIA Symposium 2002*, pages 919–923, 2002.

Index