

PROTEIN NAMES AND HOW TO FIND THEM

KRISTOFER FRANZÉN, GUNNAR ERIKSSON, FREDRIK OLSSON

*Swedish Institute of Computer Science,
Box 1263, SE-164 29 Kista, Sweden*

LARS ASKER, PER LIDÉN, JOAKIM CÖSTER

*Virtual Genetics Laboratory AB,
SE-171 77 Stockholm, Sweden*

Abstract

A prerequisite for all higher level information extraction tasks is the identification of unknown names in text. Today, when large corpora can consist of billions of words, it is of utmost importance to develop accurate techniques for the automatic detection, extraction and categorization of named entities in these corpora. Although named entity recognition might be regarded a solved problem in some domains, it still poses a significant challenge in others. In this work we focus on one of the more difficult tasks, the identification of protein names in text. This task presents several interesting difficulties because of the named entities' variant structural characteristics, their sometimes unclear status as names, the lack of common standards and fixed nomenclatures, and the specifics of the texts in the molecular biology domain in which they appear. We describe how we approached these and other difficulties in the implementation of Yapex, a system for the automatic identification of protein names in text. We also evaluate Yapex under four different notions of correctness and compare its performance to that of another publicly available system for protein name recognition.

Keywords: Knowledge; Linguistics; Natural Language Processing; Medical Information Science; Computational Molecular Biology; Information Extraction; Protein Names

1 Introduction

Terabytes of scientific data are added weekly to the pot of knowledge within the life sciences. More than 2000 completed references are added daily to MEDLINE¹ alone. Not only numerical data, but natural language text is to be taken into account when planning how to manage all this new information and knowledge. Automatic text analysis is no longer an option to strive for, but a necessity.

Linguistic knowledge and methods from computational linguistics can help in building the information access and refinement systems² that are needed to find and structure the information in the enormous amounts of scientific text produced.

Tasks that can benefit from such knowledge and methods include: the detection and extraction of names of proteins, detection of the relations between them and other substances, and the structuring, merging and refinement of that information into new knowledge.

Several areas of computational linguistics are relevant to such tasks and have matured to a point where they are ready to be exploited in real world applications.

In this paper we

- discuss the role of automatic analysis of text in a specialized domain such as molecular biology (Sections 1.1–1.3)
- discuss the nature of names in this domain and touch on the necessity of detecting named entities as a first step towards higher levels of analysis and refinement of information (Sections 1.4–1.6)
- describe a system that uses a combination of heuristic pattern matching techniques and full syntactic analysis to find names of proteins in running text (Section 2)
- discuss the general problems connected to the evaluation of such systems and propose an approach to evaluation of multi-word named entities (Sections 3.2 and 4)
- evaluate the modules in our system and compare the system with another protein name tagger on a test corpus along our proposed notions of correctness (Section 3.3).

1.1 Reading and computational text understanding

Human text understanding should be seen as an act always taking place from a certain *perspective* towards the text. In the case of information seeking, this perspective is dependent, among other things, on the background knowledge, focus, current information need, attitude, and physical and temporal constraints of the reader, and thus results in an understanding of the text that is arguably never the same as the intended understanding from the writer's point of view. Looking at it this way, it could be argued that human text understanding, when reading in the specific purpose of finding certain information, is commonly a case of partial text understanding.

Accepting this view of human text understanding, it is easy to also accept the fact that full text understanding by computers is not feasible today or

¹MEDLINE is a bibliographic database owned by the U.S. National Library of Medicine. MEDLINE can be searched via PubMed:

<http://www.ncbi.nlm.nih.gov/entrez/query.fcgi>

²For a discussion of the concepts of information access and refinement, cf., [1].

in a foreseeable future. And, in the same vein, it is still possible to build computer systems that achieve partial text understanding. Computational text understanding can then be seen as text understanding from an explicit and well defined perspective. It is limited in its scope and in its depth, but it may well be used for solving specific tasks in restricted domains. By limiting the goal — making explicit a fixed perspective — using and modeling the same constraints that influence human text understanding and reading, the usability of computer partial text understanding for a variety of tasks becomes clear.

1.2 Information access and refinement in the molecular biology domain

Tools that allow for the identification of named entities make it possible to generate annotations that can be used to index documents and document collections based on, e.g., the protein names they contain. By extending named entity recognition to other types of names such as diseases, organs and species, and by extracting the relations between such entities, directed knowledge bases can be automatically populated and used to answer questions like “What proteins in literature are associated to a certain disorder in a given organism?”.

The new high-throughput experimental procedures, such as gene expression analysis in which the expressions of multiple genes are measured simultaneously, must be validated for consistency with previous findings. By having databases of annotated documents as described above, such validation schemes can be deployed on an automatic basis. In short, the identification of multiple named entities and the relations between them can facilitate literature browsing, enhance the quality of automated experimental protocols and generate putative causative relations between genes, proteins, functions, tropism and diseases.

1.3 Information Extraction

An area of computational linguistics which focuses on text understanding from a narrow, explicit and task dependent perspective (satisfying the views in Section 1.1) is the area of Information Extraction (IE). It can be defined as the task of extracting instances of a predefined class of events (e.g., management succession events) from natural language texts, building a structured and unambiguous representation of the entities participating in these events (e.g., people, positions, companies) and the relations between them [2]. Information Extraction and its methods of evaluation have to a great extent been defined by the Message Understanding Conferences (MUCs) [3, 4, 5, 6, 7].

While Information Retrieval (i.e., document retrieval) systems aim at returning a ranked list of documents as an answer to any arbitrary information need posed in the form of a query (like search engines on the Internet), an IE system is tuned to a specific, well-specified, predefined and persistent information need. Input to the system is a stream of unrestricted text and the output is a structured representation in the form of a filled template or database record for every instance of an answer to the information need. A simplified example of the input and output of an IE system for management succession events is shown in Figure I.

Naturally, the populating of a database need not be the final goal of an information extraction system. The information detected can, for example, be used to create a summary, to create hyperlinks between information spaces to support browsing, or in any other kind of information refinement application. The area of IE is clearly related to the proposed applications in Section 1.2 and

<i>Karo Bio. Per-Olof Mårtensson has been re-appointed president after serving as chairman of the board since last spring. Mårtensson is succeeded as chairman by Bertil Hällsten, former head of S-E-Banken's pharmaceutical funds.</i>	POSITION	president
	COMPANY	Karo Bio
	IN-PERSON	Per-Olof Mårtensson
	POSITION	chairman
	COMPANY	Karo Bio
	IN-PERSON	Bertil Hällsten
	OUT-PERSON	Per-Olof Mårtensson
	POSITION	head
	COMPANY	S-E-Banken's pharmaceutical funds
	OUT-PERSON	Bertil Hällsten

Figure I: A short text and the three simplified templates it might generate in an Information Extraction system.

the experiences from the MUCs should be taken into account when developing text analysis systems for the molecular biology domain.

1.4 The importance of names

In information extraction research, it was recognized from the beginning that proper names have special significance in text, regardless of the specific task at hand; if all names in a newswire text are removed, the text loses all news value and most information in it. Because of this, one goal came to be the automatic detection, extraction and categorization of named entities³, which is a prerequisite for all higher level information extraction tasks.

For the molecular biology domain it is obvious that names of genes, proteins, chemical substances, diseases etc., are of special importance, which is why we have to begin by focusing on such entities if we want to do IE in that domain.

1.5 Named entities in molecular biology

Named entity recognition according to the traditional IE definition might be regarded a solved problem; the best MUC-participating systems have reached a performance comparable to human annotators [8]. But named entity recognition in the molecular biology domain presents a slightly different challenge because of the named entities' variant structural characteristics, their sometimes unclear status as names, and the specifics of the text domains in which they appear.

Variant structural characteristics

For several phenomena in the molecular biology domain, there are no common standards for the coining of names for newly discovered entities. Alternative names such as abbreviations and pet names are common, as are synonymous

³In the IE community, *named entities*, apart from names of people, organizations, places and products, also include monetary expressions, percentages and many kinds of temporal expressions.

names: the same entity may be referred to with different names in different research communities. Conversely, a single name may refer to several different entities as in the case of genes and proteins, where it is sometimes unclear whether the name refers to the gene or the gene product.

Apart from these characteristics, there are also very few standards to govern the construction of the words and the ways to combine them. Names may be extremely short and extremely long, both in terms of number of characters and number of words. Furthermore, the lack of explicit marking, such as e.g., capitalization, and the common inclusion of modifiers in the names make it hard to decide where a name starts and ends.

Names, are they?

The intuitive notion of what constitutes a name is easily confused when looking at words in the molecular biology domain. Often, it is hard to ascribe a position on the continuum ranging from *names* over *technical terms* to *regular noun phrases* to an arbitrary expression recurrently referring to the same specific entity. The more frequently the entity is referred to by exactly the same expression, the more name-like the expression becomes. This situation certainly holds for other text domains as well, but in this domain the liberal coining of name-like expressions and the absence of explicit markers make it difficult to separate them from the words surrounding them. It may be the case that this situation is the result of the accelerated growth of research in the field and the large number of new entities to report on in it. This together with the fact that scholars from several disciplines with different traditions separately and simultaneously are engaged in the same field makes it difficult for naming standards to evolve.

Apart from the nomenclature, there are also factors in the use of the “names” that suggest a closer relation to technical terms or regular noun phrases. There are situations in which a name-like referring expression is combined with another such expression to form a name-like reference to a third entity as well as situations when a name may be modified by one or more attributes. In some cases the resulting, larger, phrase refers to another, separate entity and in others the phrase is referring to the same entity as would the unmodified name.

The understanding of specialized text

When reading and understanding a specialized text like the scientific texts in the molecular biology domain, the notion of *perspective*, discussed above, is central. A text, with entity names with properties such as those described above, is presumably understood completely different by a domain expert compared to a layman. Some of the differences are probably due to different analysis and segmentation of the names. An expert reader’s analysis of the noun phrases *Bruton’s tyrosine kinase* and *Pasteur’s findings* would probably differ from the layman’s in a similar way. The expert would segment the first noun phrase as one lexical item, a name of a protein, while the other phrase would be analyzed as two words constituting a regular noun phrase, whereas both phrases would be considered regular noun phrases in a layman’s perspective.

A third example of the necessity of taking perspective into account is illustrated by the compound protein name *EPO mimetic peptide*. It can be analyzed as only one name, namely the whole compound, or as two names, *EPO* and the whole compound *EPO mimetic peptide*, all depending on the interest and perspective of the reader.

The more commonly addressed problem of the large amount of strange or unknown words in specialized texts is equally best seen in the light of the notion *perspective*. The words a reader already knows is a part of what constitutes his or her perspective on the text, and the interest and focus decide what words are considered strange in a particular reading.

Both the issue of segmentation of and the amount of unknown words cause problems to general linguistic analysis software. All these aspects of perspective has to be taken into account when trying to automatically analyze specialized texts.

1.6 Names of proteins

Despite the lack of common standards and fixed nomenclatures, and all the complications mentioned in Section 1.5, protein names exhibit several regularities that can be exploited in order to identify previously unseen instances. Primarily, protein names are almost always descriptive in some way. Protein characteristics such as function (e.g., *growth hormone*), localization or cellular origin (such as *HIV-1 envelope glycoprotein gp120*), physical properties (*salivary acidic protein-1*), similarities to other proteins (*Rho-like protein*) are commonly reflected in the name. Names are also constructed using a combination or abbreviation of the above. As can be noted from the examples, protein names often consist of multiple words.

It needs to be said that the definition of what should be considered a protein name is not self-evident and that it can be varied to a certain extent. In this study, we define a protein name semantically as something that denotes a single biological entity composed of one or more amino acid chains. Protein fragments or protein families are not included in this definition.

In addition to the semantic definition above, from a text structural point of view, we define a protein name as a sequence of words denoting a specific, individual protein entity. Furthermore, we also include some, more indirect, references to individual protein entities into the protein name definition, (e.g., `<prot>importin beta1</prot> derivatives`). The definition excludes non-specific reference to individuals (*transcription factor, a 89 kD protein*). It also excludes most reference to groups or classes of proteins (*protein kinases, globulins*), though phrases denoting small groups of nearly identical proteins are included (*eukaryotic RhoA-binding kinases*).

Finally, the definition of a protein name excludes anaphoric references to proteins (*this protein*).

1.7 Protein name tagging

To automatically annotate — tag — names of proteins in running text is a first step towards automatic extraction of knowledge from scientific text in the molecular biology domain. The challenge has been recognized by several research groups in recent years. Previous attempts at identifying protein names in text can be divided into systems using machine learning techniques, e.g., [9, 10], and systems based on hand-written rules, e.g., [11, 12]. The advantage of using machine learning techniques is that such a system is relatively easy to tune to new domains, provided that tagged training data exist. A hand-made system, on the other hand, requires a lot of human analysis and labor, but results in a transparent system which is easier to support, adjust and expand. Of course, mixed approaches are also possible. The system described and evaluated in this paper — Yapex — is based on hand-written rules.

2 Yapex — a protein name tagger

Arguably, building information extraction systems always involves decisions regarding how to balance recall and precision; depending on the application, one may want to focus on one or the other. Yapex initially strives for high recall with the consequence of poor precision. Later modules in the pipelined system use filtering techniques and syntactic information to boost precision, and a local dynamic dictionary is eventually applied to increase recall.

The Yapex algorithm can be described as consisting of the seven steps described in Sections 2.1–2.7 below: the first four steps are concerned with the lexical analysis of single word tokens, and the first two of these are implementations of some of the heuristic steps in the algorithm described by Fukuda et al. [11] from which the terminology of these steps is borrowed. Steps five and six are concerned with the syntactic analysis of noun phrases and of the lexical categories derived in the previous steps, and the final step utilizes the syntactic information gathered to identify new single- or multi-word protein names.

Awaiting an open source release, the Yapex system is available for testing at <http://www.sics.se/humle/projects/prothalt/>.

2.1 Lexical analysis of feature terms

Feature terms are words, e.g., *receptor* and *enzyme*, that describe the function or characteristics of a protein. These words often occur in or nearby a protein name and can be used as indicators of the presence of such a name. The analysis discriminates between internal and external feature terms, internal terms being words that belong to the name like *protein*, *particle* and *receptor*. External feature terms are words — e.g., *peptide*, *domain* and *terminal* — that act as indicators of a protein name but, most often, do not constitute a part of the name itself, according to our protein name definition. Among the internal feature terms we treat some special terms separately. These terms (*factor*, *receptor* and *enzyme*) are used as even stronger indicators of a protein name. We currently tag words as feature terms if we find them in our list of about 50 such words.

2.2 Lexical analysis of core terms

A core term constitutes the nucleus of a protein name. These terms are the parts of a protein name that show the closest resemblance to regular proper names. As candidates for these terms we pick words ending in *-ase* and *-in*, or strings with characteristics typical of protein names, i.e., strings containing instances of upper case letters or numbers, found in names of proteins like *HsMad2* and *U3-55k*. Furthermore, as all protein names do not conform to the patterns above, words are dubbed core terms if they are found in a list of established protein names such as *interferon*.

Two general filters are applied to these core term candidates to avoid over-generation: words consisting of $\geq 50\%$ non-word characters, and measuring units are discarded as core terms.

2.3 Lexical analysis of specifiers

Yapex also recognizes a third lexical category, the specifier. Specifiers are terms that often occur at the beginning or end of a protein name to, e.g., specify an individual protein. We treat Arabic and Roman numerals, single letters, Greek letter names, and combinations of these as specifiers.

2.4 Applying filters and knowledge bases

As will be seen in the evaluation (Section 3.3, Figure IV), applying the lexical analysis of the previous steps results in a large number of false hits. To remedy this low precision, the current step applies a set of lexical analysis filters. Some filters use regular expression patterns of word suffixes to rule out, e.g., names of chemical substances. Other filters use patterns of whole words/expressions to filter out bibliographical references, chemical formulas, arithmetic expressions, and amino acid sequences. A third group of pattern matching filters remove the core term annotation on words unlikely to function as core terms: words, ≥ 6 characters long consisting solely of upper case letters, or consisting of upper case letters and more than one hyphen are discarded.

Short core terms (≤ 3 characters) get special treatment. Only those found in our short-protein-name knowledge base drawn from SWISS-PROT [13] are considered core terms. All the others are tagged as potential core terms to be used later in the protein name identification process. Core terms resembling regular proper names are treated in the same way.

2.5 Finding protein name sites

To find all possible locations of protein names, this step takes advantage of the English Functional Dependency Grammar parser (ENFDG version 3.6) from Conexor Oy [14] to locate all noun phrases in the text. For every noun phrase, Yapex identifies the phrase head and its preceding lexical modifiers. This constitutes the minimal noun phrase — the noun phrase without any subordinate noun phrases — and is considered a potential protein name location.

2.6 Identifying protein names

To identify the protein name Yapex starts off by adjoining all specifiers to their preceding core, potential core, or feature term. Then all external or plural feature terms, their adjoined specifiers, and words without a lexical analysis from Yapex is stripped off from the right edge of the minimal noun phrase. From the left edge, lexical modifiers earlier identified as numerals together with measuring units are stripped off. The remaining part of the minimal noun phrase is considered a potential protein name. It is selected as such if it contains a core term, a strong feature term together with at least one other word token, a feature term with an adjoined specifier, or a potential core term together with a feature term somewhere in the full, unstripped noun phrase.

2.7 Applying a local dynamic dictionary

The relevant terms in the protein names identified in the previous step are stored in a local dictionary as regular expressions. For every document, the dictionary is used in an additional tagging pass over the text to make possible flexible matching of protein names in noun phrases undetected or misinterpreted by the ENFDG parser.

3 Evaluating a protein name tagger

Work on evaluation of protein name taggers seldom clearly specify what notions of correctness have been used when evaluating the systems, with the exception of de Bruijn and Martin [15], who present figures on *undertagging* and *overtagging*, as well as *type* and *token* matches. In this work we introduce four different notions of correctness that we have used when evaluating

the system. The different notions of correctness stress different characteristics of Yapex and the KeX system which we use as a reference system. KeX⁴ is a freely available protein name tagger based on the algorithms presented by Fukuda et al. [11].

3.1 Training and test data

From the set of answers obtained by posing the following query to MEDLINE, 99 abstracts were drawn randomly to form a reference (*training*) corpus used during development of Yapex:

```
protein binding [Mesh term] AND interaction AND
molecular
```

with the parameters *abstract, english, human, publication date 1996-2001*. The *test* corpus consists of 101 MEDLINE abstracts annotated by domain experts connected to the Yapex project. The corpus is divided into two distinct parts, the first of which contains 48 abstracts obtained as part of the result when posing the above query to MEDLINE. The first part of the test corpus contains a total of 1213 annotated protein names. The remaining 53 abstracts of the 101 in the test corpus correspond to a randomly chosen, re-tagged sub-set of the GENIA corpus [16] containing 723 annotated protein names. The reference and test corpora are mutually exclusive. The corpora are available for download at <http://www.sics.se/humle/projects/prothalt/>.

3.2 Notions of correctness

In Section 3.3 we present performance figures for Yapex and KeX on the test corpus using the following definitions of the different notions of correct matching:

SLOPPY: If any token of the proposed hit, as suggested by the tagger, matches some token of the answer key, constructed by domain experts, the hit is counted as a match.

PROTEIN NAME PARTS (PNP): Each token of the hit that matches any token of the answer key is counted as one match. This is a quantification of the SLOPPY match, that gives the degree of overlap between the proposed hit and the answer key.

STRICT: If a proposed hit matches one answer key exactly, the hit is counted as a match.

BOUNDARY:

LEFT: If a proposed hit exactly matches a left boundary in the answer key, the hit is counted as a match.

RIGHT: If a proposed hit exactly matches a right boundary in the answer key, the hit is counted as a match.

LEFT OR RIGHT: If a proposed hit exactly matches any boundary of the answer key, the hit is counted as a match.

⁴KeX can be downloaded from
<http://www.hgc.ims.u-tokyo.ac.jp/service/tooldoc/KeX/intro.html>.

3.3 Results

The goals of this evaluation are three: to show the capabilities of Yapex when run on previously unseen text; to describe the result in terms of the different notions of correctness introduced in the previous section; and to investigate how each possible combination of the filters and knowledge bases introduced in Section 2.4 and the use of the Local Dynamic Dictionary described in Section 2.7 contributes to the final result.

Comparing Yapex and KeX on previously unseen text

The first two goals of the evaluation are described in this section. To relate the performance of Yapex to previous attempts at identifying protein names in running text, we have compared Yapex to the KeX tagger.

In Table I, Yapex and KeX are compared in terms of precision, recall and F-score⁵. Looking at the SLOPPY row in the table, we can see that this is the only notion under which Yapex and KeX yield similar figures. The difference between the systems is more obvious, in favor of Yapex, when the other notions of correctness are reviewed — the figures for Yapex are substantially better when measuring the taggers’ performance in terms of PNP, STRICT, LEFT, RIGHT and LEFT OR RIGHT. We notice also that it is only under the SLOPPY condition that KeX performs close to the results it achieved in the study reported on by de Bruijn and Martin [15], but not at all close to what the KeX originators reported in Fukuda et al. [11].

	YAPEX	KeX
SLOPPY	$R = 82.1\%$	$R = 83.5\%$
	$P = 83.8\%$	$P = 82.1\%$
	$F = 82.9\%$	$F = 82.8\%$
PNP	$R = 73.7\%$	$R = 65.3\%$
	$P = 75.1\%$	$P = 44.5\%$
	$F = 74.4\%$	$F = 52.9\%$
STRICT	$R = 66.4\%$	$R = 41.1\%$
	$P = 67.8\%$	$P = 40.4\%$
	$F = 67.1\%$	$F = 40.7\%$
LEFT OR RIGHT	$R = 74.0\%$	$R = 56.2\%$
	$P = 75.5\%$	$P = 55.3\%$
	$F = 74.8\%$	$F = 55.8\%$
LEFT	$R = 71.7\%$	$R = 62.6\%$
	$P = 73.2\%$	$P = 61.5\%$
	$F = 72.5\%$	$F = 62.1\%$
RIGHT	$R = 76.3\%$	$R = 49.9\%$
	$P = 77.9\%$	$P = 49.1\%$
	$F = 77.1\%$	$F = 49.5\%$

Table I: Results for Yapex and KeX given in recall (R), precision (P), and F-score (F).

Both taggers appear to be stable in the sense that each tagger exhibits similar figures for both precision and recall in any given row in Table I, with

⁵F-score is a measure combining precision and recall:

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

where β is a parameter that represents the relative importance of Precision (P) and Recall (R), in our case equally important ($\beta = 1$).

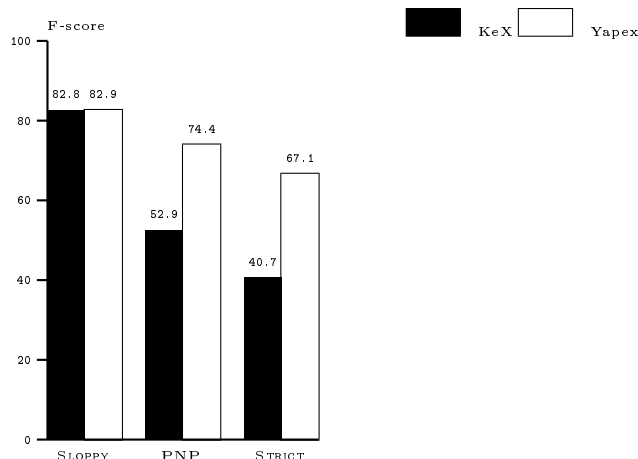


Figure II: F-score for Yapex and KeX when evaluated along the SLOPPY, PNP and STRICT notions.

one exception — the difference between recall and precision for KeX under the PNP notion. This, in combination with the results under the SLOPPY condition, suggests that KeX’ matches are too long; KeX’ high recall and precision under SLOPPY tells us that KeX’ suggestions are located close to the correct ones without too many false suggestions entirely outside. Still, KeX gives a lot of false suggestions when it comes to protein name parts.

Visualizing the F-scores in Figure II, it is clear that both a STRICT and a PNP definition of a match favors the Yapex system. The result under the PNP condition clearly shows that the overlap between the proposed hits and the corresponding answer keys is remarkably higher for Yapex than for KeX, i.e., Yapex will find more of the protein name parts. We believe that this is due to the ability of the ENFDG parser to analyze noun phrases well, and thereby predict the boundaries of protein names.

When looking at the result under the STRICT condition, the impression remains the same, suggesting that Yapex is better at finding the exact edges of the protein names. This is also shown by the result under the LEFT, RIGHT, and LEFT OR RIGHT conditions in Table I. In fact, this difference is further emphasized if we narrow the scope by looking at only the correct hits under the SLOPPY condition. Looking at the result this way (Figure III), we find that Yapex recognizes the correct left boundary in 87.4% of these cases, while the figure for recognizing the correct right boundary is a bit higher, 93%. The corresponding figures for KeX is 75% for the left boundary and 59.8% for the right. Thus, in contrast to Yapex, the KeX system appears to correctly recognize the left boundary more often than it does the right boundary. Further, given a SLOPPY hit, Yapex finds one of the left and right boundaries in 90.2% of the cases, while the same figure for KeX is 67.4%. The difference between Yapex and KeX is even greater in the case of the systems correctly matching both the left and right boundaries (i.e., STRICT) of a protein name under the SLOPPY condition; 80.9% and 49.2% for Yapex and KeX, respectively.

The impact of the filters, knowledge bases, and the Local Dynamic Dictionary

In Figure IV, there are three quadrangles illustrating the possible combinations of using filters and knowledge bases (FKB) and a Local Dynamic Dictionary (LDD) for each of the notions STRICT, PNP, and SLOPPY.

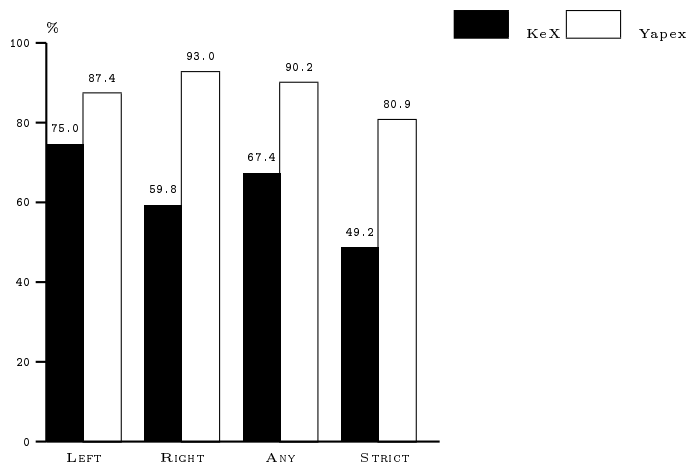


Figure III: Given a SLOPPY hit, this chart shows the probability of finding protein name boundaries for Yapex and KeX.

The way to understand a quadrangle is this: for any of the three notions in the figure, the lower left corner describes the performance of Yapex when neither filters and knowledge bases, nor the Local Dynamic Dictionary are used. The case of using Yapex with FKB, but without the LDD is represented by the upper left corner of the quadrangle. Analogously, the lower right corner denotes the use of Yapex with the LDD, but without FKB. Finally, the upper right corner represents the use of Yapex employing both FKB and LDD.

In Figure IV, we can see that the use of filters and knowledge bases promote a gain in precision, but that they at the same time contribute to lower recall. Even more interesting than the use of FKB, is the use of the Local Dynamic Dictionary. The motivation for using an LDD is to increase recall, and contrary to our intuition, precision did not drop severely even though recall increased substantially when using Yapex with an LDD.

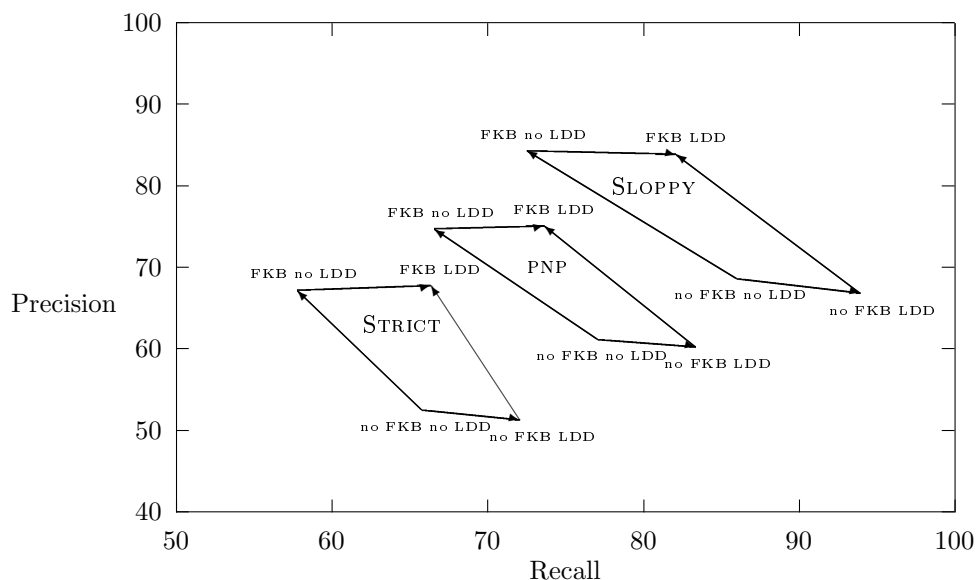


Figure IV: How the use of Filters and Knowledge Bases (FKB) and the Local Dynamic Dictionary (LDD) influences recall and precision.

4 Discussion

To problematize the metrics of recall and precision, we have chosen to evaluate along several notions of correctness. What is relevant to annotate varies with the intended application, and different methods of evaluation can highlight characteristics of competing systems. PROTEIN NAME PARTS is a relevant measure for this kind of named terminology where even human domain experts argue about the boundaries of names, since it gives an idea of how much of the multi-word proteins the systems match.

We believe that by equipping Yapex with capabilities of elaborate syntactic analysis, it performs better in recognizing protein names with respect to boundaries as well as content, than a system like KeX that does not explicitly exploit syntax. There is nothing surprising about a syntactic parser being able to aid in the detection of protein names; names cannot be found anywhere but in noun phrases. Given a perfect parser that identifies minimal noun phrases, the problem would be reduced to deciding if the noun phrase is a protein name or not. It should be noted though, that we use the ENFDG parser without modification; it has not been trained to handle this quite specific sub-domain of text. Our technique of boosting the identification of protein names by using the Local Dynamic Dictionary finds noun phrases that were not correctly analyzed as such by the parser.

What notion of correctness to actually choose to describe the performance of a protein name tagger depends on the setting in which it will be used; in one of our current applications, the tagger will be used in a browsing aid, connecting protein names in MEDLINE abstracts with the SWISS-PROT database. Since the query to SWISS-PROT can be made in a way that does not require all parts of the tagged protein name to be present in a SWISS-PROT entry to yield a match, it is not crucial that the tagger achieves perfect matches of the protein names. Thus, in our case, a figure obtained with the SLOPPY notion may suffice to describe the performance of the tagger. In an Information Extraction setting where the goal is to automatically build a high quality database, it would be more important to find the exact boundaries of the protein names, hence, such an application would benefit from a description along the STRICT or BOUNDARY notions.

A combination of the SLOPPY notion and the BOUNDARY one (as in Figure III) is good for illustrating how well a system is able to delimit a match once it has got a hold of one of the parts of the term searched for, and presenting results using PNP is suitable for highlighting the system's ability to cover multi-word names.

By using these new notions of correctness — PNP, STRICT and the variants of BOUNDARY — in addition to the commonly used SLOPPY notion, we have illustrated that it is possible to shed light on different aspects of the performance of protein name taggers. Taking into consideration the nature of protein names as such, i.e., the way they are constructed and behave, lead us to believe that the notions are suitable also for other kinds of named terminology.

It is hard to compare two systems like Yapex and KeX and still maintain a balanced record of result — there is always a risk that the test data is biased towards one of the systems. In our particular case, the domain experts that annotated the test corpus were also involved in discussing the development of Yapex, thus the annotators' definition of what constitutes a protein name is likely to favor Yapex over KeX. It is possible, e.g., that KeX' low performance under the STRICT, and especially the RIGHT condition is due to a target definition that includes parts of proteins, such as protein *sites* and *domains*. Solving problems like this calls for researchers performing similar studies in the field

to clearly state their definitions of what is considered relevant for solving a particular task. Ideally, the research community should strive for shared and open resources. The GENIA project [16] is an effort in this direction, but unfortunately, the subclasses of the GENIA protein ontology turned out to be incompatible with our definition of protein names.

Acknowledgments

Partial funding for this project has been provided by VINNOVA, the Swedish Agency for Innovation Systems.

References

- [1] Fredrik Olsson, Preben Hansen, Kristofer Franzén, and Jussi Karlgren. Information Access and Refinement — A research theme. *ERCIM News*, 46, July 2001.
- [2] Ralph Grishman. Information Extraction: Techniques and challenges. In Maria Teresa Pazienza, editor, *Information Extraction - A Multidisciplinary Approach to an Emerging Information Technology*, pages 10–27. Springer, 1997.
- [3] *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, Virginia USA, April - May 1998. Morgan Kaufmann.
- [4] *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Columbia, Maryland, USA, November 1995. Morgan Kaufman.
- [5] *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, Baltimore, Maryland, USA, August 1993. Morgan Kaufman.
- [6] *Proceedings of the Fourth Message Understanding Conference (MUC-4)*. Morgan Kaufman, June 1992.
- [7] *Proceedings of the Third Message Understanding Conference (MUC-3)*. Morgan Kaufman, May 1991.
- [8] Andrew Borthwick, John Sterling, Eugene Agichtein, and Ralph Grishman. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Proceedings of the Sixth Workshop on Very Large Corpora*, Montreal, Canada, August 1998.
- [9] Chikashi Nobata, Nigel Collier, and Jun-ichi Tsujii. Automatic term identification and classification in biology texts. In *Proceedings of the Natural Language Pacific Rim Symposium (NLPRS'2000)*, pages 369–374, November 1999.
- [10] Nigel Collier, Chikashi Nobata, and Jun-ichi Tsujii. Extracting the names of genes and gene products with a Hidden Markov Model. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING-2000)*, pages 201–207, August 2000.
- [11] Ken-ichiro Fukuda, Tatsuhiko Tsunoda, Ayuchi Tamura, and Toshihisa Takagi. Toward Information Extraction: Identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing (PSB'98)*, pages 705–716, Maui, Hawaii, January 4-9 1998.

- [12] Robert Gaizauskas, Kevin Humphreys, and George Demetriou. Information Extraction from biological science journal articles: Enzyme interactions and protein structures. In Martin G. Hicks, editor, *Proceedings of the workshop Chemical Data Analysis in the Large: The Challenge of the Automation Age*, 2001.
- [13] Amos Bairoch and Rolf Apweiler. The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucl. Acids. Res.*, 28:45–48, 2000.
- [14] Pasi Tapanainen and Timo Järvinen. A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64–71, Washington D.C., April 1997. Association for Computational Linguistics.
- [15] Berry de Bruijn and Joel Martin. Protein name tagging. Presented as a poster at the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB'00), 2000.
- [16] Nigel Collier, Hyun Seok Park, Norihiro Ogata, Yuka Tateishi, Chikashi Nobata, Tomoko Ohta, Tateshi Sekimizu, Hisao Imai, Katsutoshi Ibushi, and Jun-ichi Tsujii. The GENIA project: corpus-based knowledge acquisition and information extraction from genome research papers. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 271–272, June 1999.