An Approach to Protein Name Extraction using Heuristics and a Dictionary

Kazuhiro Seki

Laboratory of Applied Informatics Research, Indiana University, 1320 East Tenth Street, LI 011, Bloomington, Indiana 47405-3907. Email: kseki@indiana.edu

Javed Mostafa

Laboratory of Applied Informatics Research, Indiana University, 1320 East Tenth Street, LI 011, Bloomington, Indiana 47405-3907. Email: jm@indiana.edu

This paper proposes a method for protein name extraction from biological texts. Our method exploits hand-crafted rules based on heuristics and a set of protein names (dictionary). In contrast to previously proposed methods, our approach avoids the use of natural language processing tools such as part-of-speech taggers and syntactic parsers so as to improve processing speed. We implemented a prototype system for protein name extraction based on our method and conducted evaluation experiments. The result showed that our system produces outcome comparable to the state-of-the-art protein name extraction system on multiple corpora.

Introduction

Ever-growing digitized texts have resulted in a demand for automated techniques to extract novel information. Message Understanding Conferences (MUCs) (Grishman and Sundheim, 1996) represent one of the major attempts to develop information extraction (IE) techniques targeting general texts (newswire articles) in which the participants independently implement IE systems and compare their system performance on a common test set.

IE is crucial also in the field of molecular biology because of a demand for automatically discovering molecular pathways and interactions in the literature, which is, even for human experts, labor-intensive and time-consuming. Therefore, much research has been done to explore IE techniques on biological texts (Friedman et al., 2001; Ng and Wong, 1999; Proux et al., 1998; Sekimizu et al., 1998; Thomas et al., 2000).

Our ultimate goal is to realize an automated system to discover information in the biological literature, specifically, relations and interactions between specific proteins and cancer, which is expected to be beneficial for developing new medicine and treatments peculiar to cancer. To accomplish our goal, it is essential to correctly identify protein names in texts. However, automatic protein name extraction is not a trivial task. There are no common standards or fixed nomenclatures for protein names which are strictly followed in practice (Tanabe and Wilbur, 2002). As new proteins continue to be discovered and named, fixed protein name dictionaries are not necessarily helpful in extracting new protein names. Additionally, protein names often appear in shortened, abbreviated, or slightly altered forms (e.g., usage of capital and small letters and hyphens). Therefore, even the protein names that are not new and are supposed to be contained in a dictionary might be overlooked due to the way they are actually written.

This paper describes a method to identify protein names in biological texts, using hand-crafted rules based on surface clues reflecting the characteristics of protein names and a protein name dictionary. A series of evaluation experiments were conducted in terms of three criteria: accuracy, generalizability, and processing speed. The results were compared with results produced in previous work by other researchers.

Related Work

There have been numbers of attempts to develop techniques to extract protein names from biological texts. They roughly fall into three approaches, i.e., dictionary-based, rule-based, and statistical.

A dictionary used exclusively is not necessarily helpful for extracting protein names because new protein names continue to be coined and there are often some variations in the way researchers refer to identical proteins. To tackle this problem, Krauthammer et al. (2001) proposed an approach to protein and gene name extraction by using BLAST (Altschul et al., 1997), a DNA and protein sequence comparison tool. Their basic idea involves performing approximate string matching after converting both a dictionary and input texts into nucleotide sequence-like strings, which are then compared by BLAST. The results of study, however, cannot be

directly compared with our case, because they targeted both protein and gene names and the results were not separately reported.

Fukuda et al. (1998) and Olsson et al. (2002) proposed rule-based approaches. The former exploited surface clues and parts of speech. The latter, in addition to surface clues, used a syntactic parser for determining protein name boundaries. Olsson et al. conducted experiments for comparing their system (Yapex) with Fukuda's system (Kex) on 200 MEDLINE abstracts and reported that they achieved a recall of 66.4% and a precision of 67.8% on Yapex and a recall of 41.1% and a precision of 40.4% on Kex in terms of exact match.

Statistical approaches have made a considerable impact on natural language processing (NLP) research and related areas, such as part-of-speech (POS) tagging and speech recognition. In the biological domain, Collier et al. (2000) and Nobata et al. (1999) employed statistical approaches (e.g., hidden Markov models, decision trees, and probabilistic models) for detecting and classifying gene and gene product names including proteins.

On the whole, rule-based approaches have an advantage that rules can be flexibly defined and extended as needed, whereas manually analyzing targeted domain texts and creating rules are often time-consuming. Statistical approaches are relatively easy to be adapted to different domains if appropriate training corpora are provided; on the other hand, in general such approaches cannot reasonably deal with the cases that do not appear in the training corpora.

We employ a rule-based approach because of its flexibility and extensibility, as it supports refinement to adapt to newly coined protein names. In addition, the use of a protein name dictionary is explored as a complementary means.

Our Method

Previous work (Franzén et al., 2002; Fukuda et al., 1998; Nobata et al., 1999) employed POS taggers and/or syntactic parsers for detecting fragments of protein names and determining protein name boundaries based on noun phrasing. However, according to our preliminary investigation on the reference corpus made by Franzén et al. (2002), which is annotated with 1,745 protein names, most protein name fragments are nouns (85%), and thus POS taggers are unlikely to be helpful to distinguish protein names from numerous nouns. In addition, since protein name boundaries do often not correspond to noun phrase boundaries, noun phrasing based on POS taggers or parsers are not always useful to determine protein name boundaries.

Motivated by the background, our method, unlike previous work, avoids the use of NLP tools such as POS taggers and syntactic parsers, which are computationally costly. Additionally, we complementarily make use of a protein name dictionary to raise the coverage.

The rest of this section describes the details of our method. We explain how the hand-crafted rules were created and the protein name dictionary was built. In addition, an overview of our protein name extraction system, implemented based on our proposed method, is illustrated.

Protein Name Extraction by Hand-crafted Rules

Characteristics of protein names

Protein names are not easily identified because new names continue to appear and there is no fixed nomenclatures for protein names. However, there are several surface clues that can be abstracted by carefully analyzing a large number of names. These clues can be useful for detecting protein names and their fragments (Franzén et al., 2002; Fukuda et al., 1998). In the following, bold characters show the examples.

- capital letters (e.g., **ADA**, **CMS**)
- Arabic numerals (e.g., ATF-**2**, CIN**85**)
- Roman alphabets (e.g., Fc **alpha** receptor, 17**beta**-estradiol dehydrogenase)
- Roman numerals (e.g., dipeptidylpeptidase **IV**, factor **XIII**)
- frequent words appearing in protein names (e.g., myelin basic **protein**, PI 3-**kinase**, nerve growth **factor**)

We exploit the above characteristics to detect protein names and their fragments. Additionally, we also apply filters to exclude erroneous detections.

Rules for detection and filters

Our rules are based on the surface clues associated with protein names. During this stage, to avoid overlooking any potential protein names and their fragments, every word satisfying any of the following conditions is extracted as a protein name (fragment) candidate. Hereafter, we call this set of rules "*detection rules*."

- words that include capital letters (i.e., A, B, C, $\cdots$, X, Y, Z)
- words that include combinations of Arabic numerals (i.e., 0, 1, 2, 3, $\cdots$, 8, 9) and lower letter (i.e., a, b, c, $\cdots$, y, z)
- Roman alphabets that often appear as protein name fragments (i.e., alpha, beta, gamma, delta, epsilon, kappa)

- words with suffixes that often appear in protein name fragments (i.e., –nogen, –ase, –in)
- words that often appear as protein name fragments (i.e., factor(s), receptor(s))

These conditions unfortunately also misdetect words that are not protein name fragments. For example, if we extract all words containing capital letters, the words located in the beginning of sentences are inevitably extracted as protein name fragments. To decrease errors, we apply a stopword list and a set of rules (filters). The stopword list we used is the Pubmed Stopword List, which contains 133 common function words in a medical domain[1].

We set up the following rules to filter out misdetected protein name fragments. If the protein name (fragment) candidates satisfy any conditions below, they are discarded as erroneous detections.

- words that have a capital letter in the beginning followed by more than three lower letters (e.g., According, Basically)
- words that are composed of only capital letters longer than six characters. (e.g., KTPGKKKKGK)
- only one character other than 'V' (i.e., A, B, $\cdots$, U, W, $\cdots$, Y, Z)
- measuring units (e.g., nM, MM, mM, pH, MHz)
- chemical formulas (e.g., CaCl2, NH2, Ca2, HCl, Mg2)

After that, since the above rules basically detect only protein name fragments, the name boundaries of the protein name fragments have to be expanded so as to identify complete protein names. For this purpose, we used heuristic rules derived from our preliminary investigation in the reference corpus (Franzén et al., 2002).

The words and symbols matching the conditions described below are regarded as parts of protein names, and the protein name boundaries are expanded so as to include the matching words and symbols. The following are the conditions of rules that expand name boundaries rightward, where the italic characters denote the protein name fragments detected by *detection rules*, and the bold characters denote the expanded parts.

- a hyphen (optional) plus a numeral or capital letters less than three characters (e.g., *ATF* **- 2**)
- a capital letter in parentheses preceded by the protein name fragments detected by *detection rules* (e.g., *Ruk* **(I)**)

[1] `http ://www.ncbi.nlm.nih.gov/entrez/query/static/help/pmhelp.html#Stopwords`

In the meantime, conditions of the rules that expand name boundaries leftward are:

- a numeral or capital letters less than three characters plus a hyphen (optional) (e.g., **N -** *glycanase*)
- in cases where the protein name fragments detected by *detection rules* have a suffix "–in" (e.g. protein):
  - frequent words (i.e., binding, related, associated) preceded by a hyphen plus any words (e.g., parathyroid **hormone - related** *protein*)
- in cases where the protein name fragments detected by *detection rules* have a suffix "–ase" (e.g., kinase):
  - frequent suffixes (i.e., –ine, –tide, –yl) optionally followed by hyphen and numerals (e.g., **tyrosine** *kinase*)
  - frequent suffixes (i.e., –one, –sitol) optionally followed by capital letters and a hyphen (e.g., **glutathione S -** *transferase*)

Finally, the following protein name fragment candidates that are not expanded by the above rules are discarded since they are unlikely to be protein names without other qualifiers.

- protein, –ase, receptor, alpha, beta, gamma, delta, epsilon, kappa, I, II, III

A Protein Name Dictionary

In addition to the hand-crafted rules described so far, we take advantage of a protein name dictionary compiled from the SWISS-PROT (Release 40.38) and TrEMBL (Release 22.6) protein databases (O'Donovan et al., 2002) in order to extract those protein names which are not covered by the hand-crafted rules.

SWISS-PROT and TrEMBL have 120,607 and 729,579 entries for proteins respectively as of December 2002. Figure 1 shows a fragment of the SWISS-PROT protein database (TrEMBL has the same format as SWISS-PROT).

We built a protein name dictionary by extracting protein names and their synonyms from the protein name fields (`DE` in Figure 1). In the example of Figure 1, "`14-3-3-like protein GF14 chi`" is a protein name, and "`General regulatory factor 1`" is its synonym. However, since the format of the `DE` field is sometimes inconsistent, this extraction procedure produces a number of incorrect protein names, which leads to misdetection of protein names. Thus we excluded unlikely protein names which:

- begin with special characters (e.g., spaces, commas, and parentheses),

```
ID   1431_ARATH STANDARD; PRT; 267 AA.
AC   P42643; Q9M0S7;
DT   01-NOV-1995 (Rel.  32, Created)
DT   01-OCT-1996 (Rel.  34, Last sequence
     update)
DT   15-JUN-2002 (Rel.  41, Last
     annotation update)
DE   14-3-3-like protein GF14 chi
     (General regulatory factor 1).
GN   GRF1 OR AT4G09000 OR F23J3.30.
OS   Arabidopsis thaliana (Mouse-ear
     cress).
OC   Eukaryota; Viridiplantae;
     Streptophyta; Embryophyta;
     Tracheophyta;
```

Figure 1: A fragment of the SWISS-PROT protein database.

- are made up of only numerals or less than two characters, and
- end with a hyphen or apostrophe.

In addition, we excluded the protein names composed of less than three tokens (words and symbols) since we found that these protein names harmed the system accuracy on the whole. This is presumably because shorter protein names are more general and not necessarily protein names depending on the context.

Furthermore, we applied the *detection rules* on the initial set of protein names extracted from SWISS-PROT and TrEMBL, and the protein names detected by the rules were not included in the dictionary, so as to decrease dictionary size and improve processing speed.

As a result, 114,876 protein names and their synonyms were included in the dictionary.

An Implementation

We implemented a prototype system for protein name extraction based on the rules and the dictionary described above. The system was coded the Perl programming language. Figure 2 illustrates an overview of our system.

In the initial step, a biological text is given to our system as an input and is preprocessed. Specifically, the input text is partitioned into sentences and then tokenized, where tokens are defined as words and symbols. For instance, `PI 3-kinase` will be separated into four tokens, i.e., `PI`, `3`, `-`, and `kinase`. Then, protein name fragment candidates are detected based on the hand-crafted rules, followed by a filter to rule out misdetected protein name fragments. After that, protein name boundaries of the detected fragments are expanded both rightward and leftward. Then, another filter is applied to exclude erroneous detections. Lastly, a protein name dictionary is complementarily consulted for extracting those protein names which are not covered by the preceding rule-based detection module.

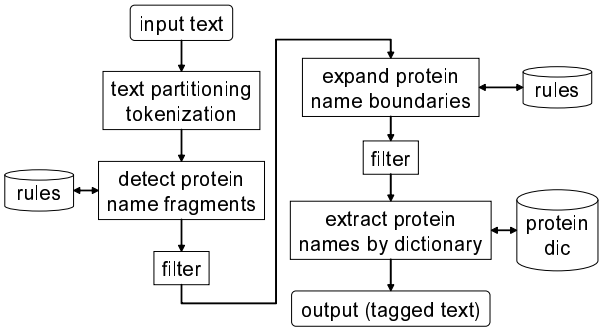

Figure 2: An overview of our protein name extraction system.

Evaluation

Overview

To evaluate the effectiveness of our method, we conducted a series of comparative experiments, in which our system was compared with Yapex (Franzén et al., 2002; Olsson et al., 2002) in terms of accuracy, generalizability, and processing speed. Yapex is the state-of-the-art protein name extraction system based on hand-crafted rules, and the system is currently available through a CGI program on the Web[2].

We used the same annotated corpora as Olsson et al (2002) used for their system evaluation[3]. The corpora consist of reference corpus and test corpus, which are composed of 99 and 101 MEDLINE abstracts, respectively. In the test corpus, 53 abstracts are a subset of the GENIA corpus (Ohta et al., 2002), but they were re-tagged by domain experts according to their definition of protein names. Notice that we used only the reference corpus in developing our system.

In the description of our experiments below, *rule* denotes a version of our system exclusively relying on the hand-crafted rules, and *rule+dic* denotes another version of our system using a dictionary as well as the rules.

Accuracy

Precision, recall, and F-score are used as evaluation metrics. Precision is the number of protein names a system correctly detected divided by the total number

[2] `http://www.sics.se/humle/projects/prothalt/`

[3] This particular study was selected for comparison because according to our survey it has produced the best protein name extraction results so far.

of ones the system detected. Recall is the number of protein names a system correctly detected divided by the total number of ones contained in an input text. F-score combines recall and precision into a single score and is defined as in Equation (1).

$$F\text{-}score = \frac{2 \times precision \times recall}{precision + recall} \quad (1)$$

For judgment of correctness, we use three criteria: exact, partial, and fragment matches. As for exact match, every fragment composing a protein name has to be detected correctly, whereas, for partial match, a detected protein name is counted as correct in the case where *any* fragments composing the protein name are correctly detected. For fragment match, the counting unit is fragments; that is, each fragment composing a protein name is to be judged independently whether it is correctly detected or not.

Generalizability

Generalizability is another important criterion to evaluate our system because we aim at extracting comprehensive information related to proteins for our ultimate purpose, regardless of the source of documents. Although annotation policies or definition of protein names could be different among different corpora annotated with protein names, it is desirable to achieve constant performance even on different corpora.

To evaluate the generalizability of our system, an experiment was conducted on another annotated corpus, the GENIA corpus version 3.0 (Ohta et al., 2002), which consists of 2,000 MEDLINE abstracts and is annotated with a subset of the substances and the biological locations involved in reactions of proteins. We regarded a subset of protein as protein names according to their ontology.

As for evaluation metrics, we used precision, recall, and F-score described above.

Processing speed

Our system is supposed to be faster than the previous work that incorporate NLP tools, such as POS taggers and syntactic parsers, since we avoid using them. To show our advantage, we compared our system with the Yapex protein name extractor in terms of processing speed. As Yapex is available only through a web-based CGI program at this moment, we also made our system available on the Web as a CGI program and compared the performance. That is, we measured the time that each system needed to return a result after an input text was submitted. Although computing performance of web servers is expected to be different and it does include data transmission time and so on, these experiments were conducted to provide a rough idea of comparative processing speed.

We used the first 15 MEDLINE abstracts of the reference corpus as an input text, which is made up of 3,000 words.

As computer and network loads are usually different during different time of the day, in order to decrease the influence of this factor we accessed each system three times each hour for a day (24 hours) and selected the minimum turn around times for comparison.

Results and Discussion

Accuracy

Table 1 and Table 2 shows the results of the experiment concerning accuracy, where the figures in the column "Yapex" are directly cited from their paper (Olsson et al., 2002), and "*rule*" and "*rule+dic*" denote our system.

Table 1: A comparison between Yapex and our system on the reference corpus.

evaluation criteria		Yapex	*rule*	*rule+dic*
exact	precision	67.2	57.3	57.9
	recall	67.1	74.8	76.2
	F-score	67.1	64.9	65.8
partial	precision	82.9	70.1	70.0
	recall	82.8	91.4	92.1
	F-score	82.9	79.3	79.5
fragment	precision	73.4	68.6	67.8
	recall	74.0	75.7	79.4
	F-score	73.7	72.0	73.1

Table 2: A comparison between Yapex and our system on the test corpus.

evaluation criteria		Yapex	*rule*	*rule+dic*
exact	precision	68.8	57.6	58.3
	recall	65.3	65.3	66.8
	F-score	67.0	61.2	62.3
partial	precision	85.3	77.3	77.3
	recall	80.9	87.6	88.6
	F-score	83.0	82.2	82.6
fragment	precision	77.8	73.6	73.6
	recall	73.3	66.9	71.2
	F-score	75.5	70.1	72.3

In the case where the protein name dictionary was used in addition to the rules, the recall improved in the range of 0.7–4.3 points regardless of the corpora. This result demonstrated that the dictionary was beneficial

to detect the protein names uncovered by the handcrafted rules exploiting surface clues. Moreover, using the dictionary marginally improved precision as well in several cases.

Incidentally, in the case where only the dictionary was applied (which is not included in Table 1 and Table 2), precision and recall in terms of exact match were 42.6% and 1.7% for the reference corpus and 49.1% and 1.4% for the test corpus, respectively. Their recall are significantly low because we included only the protein names composed of more than four tokens in our dictionary. Although we may add shorter protein names in our dictionary and can increase the recall, it deteriorates the overall performance of our system.

When compared to Yapex, our system generally obtained lower precision than Yapex irrespective of the use of the protein name dictionary; whereas, our system mostly outperformed Yapex in terms of recall. Consequently, the F-scores of our system were found to be quite comparable with those of Yapex, despite the fact that our system does not incorporate syntactic parsers as used in Yapex.

We evaluated our system on several criteria, i.e., exact, partial, and fragment matches and precision, recall, and F-score. Which criterion is important depends on what purpose we use the system for. Considering our ultimate goal, that is, IE for the cancer-protein interaction, exact match is important for distinguishing each protein name and extracting information associated with them. Since our system achieved a high recall of 92.1% and 88.6% on partial match, more specific rules for expanding protein name boundaries and for filtering out erroneous detections are crucial in order to improve the performance on exact match.

Generalizability

Table 3 shows the results of protein name extraction on the GENIA corpus (version 3.0). We also show the result on Yapex for comparison.

Table 3: A comparison between Yapex and our system in terms of accuracy on the GENIA corpus (version 3.0).

evaluation criteria		Yapex	*rule*	*rule+dic*
exact	precision	45.3	41.7	42.6
	recall	44.0	52.9	54.4
	F-score	44.7	46.7	47.8
partial	precision	67.1	58.8	59.1
	recall	65.1	74.7	75.6
	F-score	66.1	65.8	66.4
fragment	precision	61.5	65.5	65.9
	recall	54.5	58.6	61.6
	F-score	57.8	61.8	63.7

Our system's precision on fragment match outperformed that of Yapex, but the tendency of that Yapex producing higher precision and our system producing higher recall holds as a whole. The performance of both Yapex and our system dropped by around 10–20 points in F-score as compared with the experiments on the other corpora shown in Table 1 and Table 2. This was caused by the difference of protein ontology between Franzén et al. (2002) and Ohta et al. (2002). An analysis on the GENIA corpus is needed for further improvement.

Processing Speed

Firstly, we submitted a null input (no test set provided) to each system and measured their turn around times to establish approximate difference of transmission times related to the two systems. We measured them five times for each system and selected the minimum. The result is shown in Table 4.

Table 4: Turn around times with a null input.

Yapex	*rule*	*rule+dic*
1.76 sec	1.35 sec	1.39 sec

Secondly, we evaluated each system with an input text of 3,000 words. Table 5 shows their turn around times for protein name extraction, which were measured on 29–30, December 2002.

Table 5: Turn around times with an input text of 3,000 words.

Yapex	*rule*	*rule+dic*
11.11 sec	2.84 sec	3.99 sec

Although the turn around times cannot be directly compared due to other factors, the result indicates that our system was around 3–4 times faster than Yapex.

Additionally, in the case where our system processed the input text of 3,000 words not through a web-based CGI program but via a stand-alone program, it completed the process by 1.38 seconds (*rule*) and 2.62 seconds (*rule+dic*). This indicates that 1.46 (= 2.84−1.38) seconds and 1.37 (= 3.99 − 2.62) seconds account for the overheads of the process through the CGI program, respectively. These times approximately correspond to the turn around times shown in Table 4. Assuming that this also holds in the case of Yapex, its processing time will be around 9.35 (= 11.11 − 1.76) seconds, implying that our system is about 4–7 times faster than Yapex.

Conclusions and Future Work

In this paper, we presented a method for extracting protein names in biological texts, which is mainly based on simple knowledge reflecting characteristics of protein names. We also take advantage of a protein name dictionary as a complementary means to cover the protein names that our rules cannot identify. Our method, as opposed to the previous work, does not use POS taggers and/or syntactic parsers, which are computationally expensive, since the information that those NLP tools provide is not necessarily helpful for protein name extraction. We implemented a prototype system based on our proposed method and conducted comparative experiments. The results demonstrated that our method is comparable to the Yapex protein name extractor which uses a syntactic parser. Further, although they cannot be directly compared, our experiment implied that our system appeared to be 4–7 times faster than Yapex.

Future work would include further refinements of the hand-crafted rules for expanding protein name boundaries and improvements of the generalizability of our method.

References

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402.

Collier, N., Nobata, C., and Tsujii, J. (2000). Extracting the names of genes and gene products with a hidden markov model. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 201–207.

Franzén, K., Eriksson, G., Olsson, F., Asker, L., and Lidén, P. (2002). Exploiting syntax when detecting protein names in text. In *Workshop on Natural Language Processing in Biomedical Applications*.

Friedman, C., Kra, P., Yu, H., Krauthammer, M., and Rzhetsky, A. (2001). GENIES: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17 Suppl. 1:S74–S82.

Fukuda, K., Tsunoda, T., Tamura, A., and Takagi, T. (1998). Toward information extraction: Identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 3, pages 705–716.

Grishman, R. and Sundheim, B. (1996). Message Understanding Conference - 6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 466–471.

Krauthammer, M., Rzhetsky, A., Morozov, P., and Friedman, C. (2001). Using BLAST for identifying gene and protein names in journal articles. *GENE*, (259):245–252.

Ng, S. and Wong, M. (1999). Toward routine automatic pathway discovery from on-line scientific text abstracts. In *Proceedings of Genome Informatics*, volume 10, pages 104–112.

Nobata, C., Collier, N., and Tsujii, J. (1999). Automatic term identification and classification in biology texts. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium*, pages 369–374.

O'Donovan, C., Martin, M. J., Gattiker, A., Gasteiger, E., Bairoch, A., and Apweiler, R. (2002). High-quality protein knowledge resource: SWISS-PROT and TrEMBL. *Brief Bioinform*, 3(3):275–284.

Ohta, T., Tateisi, Y., Kim, J.-D., Mima, H., and Tsujii, J. (2002). The GENIA corpus: an annotated research abstract corpus in molecular biology domain. In *Proceedings of the Human Language Technology Conference*.

Olsson, F., Eriksson, G., Franzén, K., Asker, L., and Lidén, P. (2002). Notions of correctness when evaluating protein name taggers. In *Proceedings of the 19th International Conference on Computational Linguistics*.

Proux, D., Rechenmann, F., and Julliard, L. (1998). Detecting gene symbols and names in biological texts: A first step toward pertinent information extraction. In *Proceedings of Genome Informatics*, volume 9, pages 72–80.

Sekimizu, T., Park, H. S., and Tsujii, J. (1998). Identifying the interaction between genes and gene products based on frequently seen verbs in MEDLINE abstracts. In *Proceedings of Genome Informatics*, volume 9, pages 62–71.

Tanabe, L. and Wilbur, J. (2002). Tagging gene and protein names in full text article. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain*, pages 9–13.

Thomas, J., Milward, D., Ouzounis, C., Pulman, S., and Carroll, M. (2000). Automatic extraction of protein interactions from scientific abstracts. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 5, pages 538–549.