A Probabilistic Model for Identifying Protein Names and their Name Boundaries

Kazuhiro Seki and Javed Mostafa
Laboratory of Applied Informatics Research, Indiana University
1320 East Tenth Street, LI 011, Bloomington, Indiana 47405-3907
{kseki,jm}@indiana.edu

Abstract

This paper proposes a method for identifying protein names in biomedical texts with an emphasis on detecting protein name boundaries. We use a probabilistic model which exploits several surface clues characterizing protein names and incorporates word classes for generalization. In contrast to previously proposed methods, our approach does not rely on natural language processing tools such as part-of-speech taggers and syntactic parsers, so as to reduce processing overhead and the potential number of probabilistic parameters to be estimated. A notion of certainty is also proposed to improve precision for identification. We implemented a protein name identification system based on our proposed method, and evaluated the system on real-world biomedical texts in conjunction with the previous work. The results showed that overall our system performs comparably to the state-of-the-art protein name identification system and that higher performance is achieved for compound names. In addition, it is demonstrated that our system can further improve precision by restricting the system output to those names with high certainties.

1 Introduction

Ever-growing digitized texts have resulted in a demand for automated techniques to extract novel information from texts. Message Understanding Conferences (MUCs) [10] represent one of the major attempts to develop information extraction (IE) techniques targeting general texts (newswire articles) in which the participants independently implemented IE systems and compared their system performance on a common test set.

IE is crucial also in the field of cellular and molecular biology because of a strong 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 conducted to explore IE techniques on biomedical texts [1, 7, 8, 11, 15, 18, 19, 21, 23].

Our ultimate goal is to realize an automated system to discover novel information in the biomedical 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, we start with identifying protein names appearing in biomedical texts. However, automatic protein name identification is not a trivial task. This is partially because there are no common standards or fixed nomenclatures for protein names that are followed in practice [4]. As new proteins continue to be discovered and named, predefined protein name dictionaries are not necessarily helpful in identifying new protein names. Additionally, protein names frequently appear in shortened, abbreviated, or slightly altered forms (e.g., the use of capital and small letters and hyphens). Therefore, even the protein names that are already known and are supposed to be contained in a dictionary might be overlooked due to the way they are actually written. Another challenging issue for identifying protein names is to find their name boundaries. According to our preliminary research on 99 MEDLINE abstracts, 42% of protein names are composed of multiple tokens (tokens are defined as words and symbols), and these tokens include common nouns, adjectives, adverbs, and even conjunctions, which makes it difficult to distinguish protein names from the surrounding texts [22].

We propose a statistical approach to identifying protein names in biomedical texts. Our approach employs probabilistic models for finding protein name boundaries and for restricting the final output to those with high certainties so as to improve the accuracy of identification. The probabilistic models exploit surface clues that reflect the characteristics of protein names. To evaluate our method, a series of experiments is conducted to compare results with previous findings by other researchers.

Section 2 briefly summarizes past work related to pro-

Proceedings of the Computational Systems Bioinformatics (CSB'03)
0-7695-2000-6/03 $17.00 © 2003 IEEE

tein name identification, and Section 3 details our proposed method. In Section 4, the methodology of evaluation is described and the result is presented and discussed. In Section 5 and Section 6, we conclude this paper with our findings and future work.

2 Related Work

There have been number of attempts to develop techniques to identify protein names in the biomedical literature. They roughly fall into three approaches, that is, dictionary-based, heuristic rule-based, and statistical.

A technique based exclusively on a dictionary is not necessarily helpful for identifying protein names because new protein names continue to be created and there are often many variations in the way identical proteins are referred to. To tackle this problem, Krauthammer et al. [13] proposed an approach to protein and gene name extraction, using BLAST [2], a DNA and protein sequence comparison tool. Their basic idea involves performing approximate string matching after converting both dictionary entries and input texts into nucleotide sequence-like strings, that then can be compared by BLAST. The results they reported, 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. [9], Narayanaswamy [14], and Olsson et al. [17] proposed rule-based approaches. They exploited surface clues for detecting protein name fragments (i.e., parts of protein names) and used a part-of-speech tagger and/or a syntactic parser for finding protein name boundaries. Typically, the surface clues include the following features, where bold characters indicate the corresponding 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**)
- Words appearing frequently in protein names (e.g., myelin basic **protein**, PI 3-**kinase**, nerve growth **factor**)

Olsson et al. [17] conducted experiments that compared their system (Yapex) with Fukuda's system (Kex) on 101 MEDLINE abstracts. Yapex achieved a recall of 61.0% and a precision of 62.0% as compared to a recall of 37.5% and a precision of 34.3% on Kex in terms of exact match. Incidentally, Narayanaswamy et al. [14] reported to have achieved a recall of 69.1% and a precision of 96.9% on 55 MEDLINE abstracts, where the precision is much higher than both Yapex and Kex. Notice that, however, Narayanaswamy et al. were targeting both protein and gene names and did not distinguish them in the evaluation.

Statistical approach has made a considerable impact on natural language processing (NLP) research and related areas, such as part-of-speech (POS) tagging, parsing, and speech recognition. In the bioinformatics domain, Collier et al. [3], Nobata et al. [16], and Kazama et al. [12] employed statistical approaches (e.g., hidden Markov models, decision trees, and support vector machines) for detecting and classifying gene and gene product names including proteins. The features used in their methods are mostly the same as those used in rule-based approaches, that is, surface clues and parts of speech.

Comparing rule-based and statistical approaches, rule-based approaches have an advantage in a sense that rules can be flexibly defined and extended as needed, but manually analyzing targeted domain texts and crafting rules are often time-consuming. Statistical approaches are relatively easy to be applied if appropriate models and training data are provided. However, manually creating training data (i.e., corpora annotated with protein names in this case) is also time-consuming and needs biomedical expertise, and an insufficient amount of training data leads to the data sparseness problem. In general, to achieve higher performance, more complex models are needed, which, however, often require more training data in order to reasonably estimate the increasing number of parameters.

We mainly employ a statistical approach using a probabilistic model for identifying protein names with an emphasis on finding name boundaries. Our method solely exploits surface clues, unlike previous work, avoiding the use of POS taggers and syntactic parsers. According to our preliminary investigation on the corpus annotated with 1,745 protein names made by Franzén et al. [6], protein name fragments can be not only nouns but also adjectives, adverbs, verbs, and conjunctions, and thus POS tags are not necessarily helpful to detect protein names and their boundaries. Avoiding the use of such NLP tools will reduce processing overhead and the potential number of parameters to be estimated. Moreover, we generalize words composing protein names to word classes and also apply a smoothing method in order to compensate for the limited amount of training data.

3 Our Method

3.1 Overview

Figure 1 depicts an overview of our protein name identification system based on the method to be described in

this section. In the preprocessing module, an 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`.

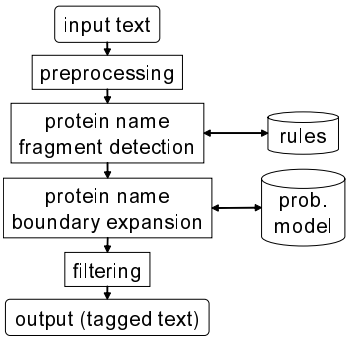

Figure 1. An overview of our protein name extraction system.

Then, we identify protein names through three steps. First, protein name fragments are detected by heuristic rules relying on surface clues which are commonly used for protein name identification. Second, for each of the detected protein name fragments, its name boundary is expanded based on a probabilistic model to locate complete protein name candidates. Lastly, a filter is applied to the candidates so as to exclude erroneous detections, and only those with high certainties are output. Each step is further explained in Section 3.2–Section 3.4.

3.2 Protein name fragment detection

We use several heuristic rules to detect protein name fragments which have been commonly used in previous studies [6, 9, 14, 17, 20]. Words which satisfy any of the following conditions are detected as potential protein name fragments.

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

These conditions unfortunately also detect words that are not protein name fragments. For example, if we extract all words containing capital letters, words located in the beginning of sentences are to be inevitably extracted as protein name fragments. To decrease errors, we exclude the following tokens:

- Words with a capital letter in the beginning followed by more than three lower case letters (e.g., According, Basically)
- Words composed of only capital letters longer than six characters. (e.g., KTPGKKKKGK)
- Only one character (i.e., A, B, $\cdots$, Y, Z)
- Measuring units (e.g., nM, MM, mM, pH, MHz)
- Chemical formulas (e.g., CaCl2, NH2, Ca2, HCl, Mg2)
- Words included in a stopword list. In this study, we used the Pubmed Stopword List, which contains 133 function words[1].

3.3 Protein name boundary expansion

We employ a probabilistic model for expanding/finding protein name boundaries leftward and rightward for each of the detected protein name fragments. Below, we will explain the details of our model, focusing on expanding name boundaries rightward as an example.

Let w_i denote one of the protein name fragments detected in the previous initial detection step (see Section 3.2). Given a fragment w_i, the probability that a token w_{i+1} following to w_i is also a protein name fragment can be expressed as a conditional probability $P_p(w_{i+1}|w_i)$, assuming a first-order Markov process. Likewise, the probability that w_{i+1} is *not* a protein name fragment is to be expressed as $P_n(w_{i+1}|w_i)$.

We expand protein name boundaries based on these probability estimates. In the case where there is not a name boundary between w_i and w_{i+1} (i.e., w_{i+1} is also a protein name fragment), $P_p(w_{i+1}|w_i)$ is expected to be greater than $P_n(w_{i+1}|w_i)$. Thus, we regard w_{i+1} as a protein name fragment if the following condition holds:

$$P_p(w_{i+1}|w_i) > P_n(w_{i+1}|w_i) \quad (1)$$

However, estimating these probabilistic parameters requires a large amount of training texts annotated with protein names, which are labor-intensive to create. To make matters worse, simply using a large-scale corpus cannot be

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

a substantial solution due to the characteristics of protein names: new protein names continue to be created. Previously unseen data are fatal for probability estimation such as maximum likelihood estimation.

To reduce the influence of the data sparseness problem, we generalize words (tokens) to word classes as shown in Table 1. They are automatically uniquely assigned to each word (token).

class	examples
suffix_in	protein, oncoprotein, lactoferrin
suffix_ase	kinase, transferase, peptidase
word	the, a, an
acronym	CN, TrkA, USF
arabic_num1	1, 2, 3
arabic_num2	12, 76, 32
roman_num	I, II, III
roman_alpha	alpha, beta, gamma
punctuation	comma (,), period (.)
symbol	), (, %, +

Table 1. Examples of word classes.

In Table 1, suffix classes (e.g., *suffix_in*) are dynamically generated by extracting a sequence of a vowel, one or more consonants (if any), and either a vowel or a consonant in the end of a word in question. However, class *word* will be assigned in the case where the resulting suffix is equal to or longer than the remainder of the word in length, so as to prevent inaccuracies in extraction of suffixes.

Integrating the word classes to the probabilistic model, we define bigram class models as in Equation (2), where c_i denotes the word class of w_i.

$$\begin{aligned} P_p(w_{i+1}|w_i) &= P_p(w_{i+1}|c_{i+1}) \cdot P_p(c_{i+1}|c_i) \\ P_n(w_{i+1}|w_i) &= P_n(w_{i+1}|c_{i+1}) \cdot P_n(c_{i+1}|c_i) \end{aligned} \quad (2)$$

The probabilistic parameters shown in Equation (2) can be estimated based on corpora annotated with protein names. However, the models still contain raw words w_{i+1}, which are likely to cause the data sparseness problem. To avoid it, we use Witten-Bell smoothing [24] in estimating the probability of having the word w_{i+1} from the class c_{i+1}, which we found to perform better.

Similarly, we adopt this model to expand/find protein name boundaries leftward as well. The probability functions are defined as in Equation (3), where w_{i-1} and c_{i-1} denote the token preceding to the detected protein name fragment w_i and its word class, respectively.

$$\begin{aligned} P_p(w_{i-1}|w_i) &= P_p(w_{i-1}|c_{i-1}) \cdot P_p(c_{i-1}|c_i) \\ P_n(w_{i-1}|w_i) &= P_n(w_{i-1}|c_{i-1}) \cdot P_n(c_{i-1}|c_i) \end{aligned} \quad (3)$$

3.4 Filtering

Our ultimate goal is to automatically extract novel information associated with proteins and cancer from the literature, where protein name identification is a fundamental element whose performance will strongly affect the rest of the IE process. Although high recall and high precision are ideal, there is a trade-off between the two measures. In this context, it is desirable that we could choose which measure is preferred (i.e., high recall with low precision, high precision with low recall, or balanced), according to the purpose. This can be done by restricting the system output based on some certainty measure that indicates the extent to which the detected protein names are likely to be actual protein names.

We are currently using certainty score $C(\cdot)$ defined as in Equation (4), where $w_1 \cdots w_n$ denotes a sequence of tokens detected as a protein name, and $F(x)$ and $F_p(x)$ denote a frequency of x in training data and a frequency of x which appears as a protein name fragment, respectively.

$$\begin{aligned} C(w_1 \cdots w_n) &= \frac{1}{n}\sum_{i=1}^{n} P_c(w_i) \\ \text{where } P_c(w_i) &= \begin{cases} \dfrac{F_p(w_i)}{F(w_i)} & if\ F(w_i) \geq 3 \\ \dfrac{F_p(c_i)}{F(c_i)} & otherwise \end{cases} \end{aligned} \quad (4)$$

$P_c(w_i)$ indicates a degree to which word w_i is a protein name fragment, and $C(w_1 \cdots w_n)$ is an average of all probabilities for w_1 to w_n. Probability $P_c(w_i)$ will take a high value in the case where a token w_i is predominantly used as a protein name fragment in training texts since $F_p(w_i)$ approaches to $F(w_i)$. Additionally, in the case where the frequency of w_i is small, instead we use the frequency of its word class because low frequent data are statistically less reliable. We set the cutoff to 3.

The word classes used in computing the certainty score are basically the same as those used in protein name boundary expansion shown in Table 1. However, only acronyms are treated differently: more specific word classes are given. For instance, `HsMad1` will be associated with word class *acronym_AaAa*. The suffix of the class, *AaAa*, is derived as follows: consecutive capital letters, small letters, and numbers are squeezed into one character *A*, *a*, and *0*, respectively; and then, if any, *0* in the end of strings is stripped. The assumption for this transformation is that protein name acronyms have some patterns in the usage of capital letters, small letters, and numbers.

4 Evaluation

4.1 Overview

To evaluate the effectiveness of our approach, we implemented a protein name identification system based on the probabilistic models described in Section 3 and conducted a series of experiments, in which our system was compared with the Yapex protein name identification system [6, 17].

There are three reasons this particular study was selected for comparison. According to our survey, Yapex is one of the state-of-the-art protein name identification systems, which is based on hand-crafted rules, the system is publicly available through a CGI program on the *Proteinhalt i text* (protein concentration in text) project homepage [5], and the annotated corpora used for Yapex's evaluation are also publicly available.

As mentioned above, we used the same corpora as Franzén et al. [6] and Olsson et al. [17] used for Yapex's evaluation. The corpora consist of a reference corpus and a test corpus with 99 and 101 MEDLINE abstracts, respectively. The reference corpus, which is annotated with 1,745 proteins, was used for training our probabilistic models and the test corpus, which is annotated with 1,966 protein names, was used for evaluation.

4.2 Evaluation measures

Precision, recall, and F-score are used as evaluation measures. Precision is the number of protein names a system correctly detected, divided by the total number of protein names detected by the system. Recall is the number of protein names a system correctly detected, divided by the total number of protein names contained in the input text. F-score combines these measures, i.e., recall and precision, into a single score and is defined as in Equation (5).

$$F\text{-}score = \frac{2 \times precision \times recall}{precision + recall} \tag{5}$$

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 correctly detected to be judged as correct, 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 a fragment; that is, *each* fragment composing a protein name is to be judged independently whether it is correctly detected or not.

4.3 Results and discussion

Overall performance

Table 2 shows the result of the comparative experiment. The values in the column "Yapex" are directly cited from the *Proteinhalt i text* project homepage [5], and "*Prob*" denotes our system based on the probabilistic models. A threshold for the certainty score (see Section 3.4) was set to 0.245 in this experiment, which was derived by applying two-fold cross-validation to the training data so as to maximize F-score for exact match. To put it more precisely, we divided the training data into two sets of text *A* and *B* in an equal size, and used *A* for computing certainty scores for *B* and, in turn, used *B* for computing certainty scores for *A* with varying the threshold. Then we took a mean of the thresholds which maximized F-score for each set.

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

evaluation criteria		Yapex	*Prob*
exact	recall	59.9	66.9
	precision	62.0	60.1
	F-score	61.0	63.3
partial	recall	81.4	86.0
	precision	84.3	77.2
	F-score	82.8	81.4
fragment	recall	76.2	75.6
	precision	75.8	74.3
	F-score	76.0	75.0

When compared to Yapex, our system obtained about 2–7 points lower precision irrespective of the criteria for judgment of correctness (i.e., exact, partial, and fragment matches), while our system mostly outperformed Yapex in terms of recall. Consequently, the F-scores of our system were found to be quite comparable to those of Yapex, despite the fact that our method does not rely on POS taggers or syntactic parsers as used in Yapex.

We evaluated our system on several criteria, i.e., exact, partial, fragment matches and recall, precision, 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 would be important for distinguishing number of protein names and associating extracted information with them. Incidentally, high recall would be preferable in the case where comprehensive information is needed, while high precision would be desirable in the case where the reliability of information is important. We will show later in this section that higher precision can be achieved by varying a threshold for the cer-

tainty score.

Alternative models

For generalization, we made use of a bigram class model, which is an integration of class-class and class-word transition probabilities. To verify the effectiveness of the model, we compared it with two other alternatives: one without word classes (less generalized) and one without words (more generalized). As an example, Equation (6) and Equation (7) show the conditions of these models for expanding name boundaries rightward. Notice that Equation (6) is identical to Equation (1).

$$P_p(w_{i+1}|w_i) > P_n(w_{i+1}|w_i) \quad (6)$$

$$P_p(c_{i+1}|c_i) > P_n(c_{i+1}|c_i) \quad (7)$$

Table 3 shows the results of protein name identification using our proposed model and the alternatives, in which "*word*" and "*class*" denote the word transition model defined as in Equation (6) and the class transition model as in Equation (7), respectively.

Table 3. A comparison between our proposed model and word transition and class transition models on the test corpus.

evaluation criteria		*Prob*	*word*	*class*
exact	recall	66.9	41.8	41.6
	precision	60.1	41.0	48.0
	F-score	63.3	41.4	44.6
partial	recall	86.0	74.7	72.0
	precision	77.2	73.3	83.0
	F-score	81.4	74.0	77.1
fragment	recall	75.6	48.0	63.8
	precision	74.3	67.2	66.1
	F-score	75.0	56.0	64.9

In terms of partial match, there is less difference among these three models. This is expected by the definition of partial match: detected protein names are judged as correct if any token contained in them is correctly detected. In other words, if any fragment of protein names is correctly detected by hand-crafted rules in the initial detection phase, it is regarded as correct; that is, name boundary expansion does not have much influence on the accuracy of partial match.

For other evaluation criteria, our proposed model outperformed the others, especially in exact match, which indicates that our model is appropriately generalized and effectively incorporates word classes for protein name identification.

Performance for compound terms

Since our method is focusing on name boundary expansion using word (class) transition based on bigrams, our method is expected to be more effective particularly for compound protein names. To demonstrate the advantage, we evaluated Yapex and our system solely on compound protein names. The test corpus used here is the same as the one used above (i.e., Yapex test corpus) and contains 897 compound protein names. Table 4 shows the result, in which Yapex's result was obtained by submitting the test corpus to the Yapex demo page [5] on March 27, 2003.

Table 4. A comparison between Yapex and our system for compound protein names on the test corpus.

evaluation criteria		Yapex	*Prob*
exact	recall	53.2	59.0
	precision	49.7	63.7
	F-score	51.4	61.2
partial	recall	73.3	73.5
	precision	68.5	79.3
	F-score	70.8	76.3
fragment	recall	65.8	65.0
	precision	65.1	76.8
	F-score	65.4	70.4

In the case where only compound protein names are considered, irrespective of evaluation criteria, our system greatly outperformed Yapex especially in exact match. This result shows that our proposed probabilistic model is fairly effective in expanding and finding name boundaries for compound protein names, and that the word classes used for generalization efficiently capture the characteristics of protein names to a large extent.

Filtering based on certainty

Lastly, the effectiveness of the certainty score introduced in Section 3.4 was evaluated. We varied a threshold for the certainty score, so as to draw a recall-precision curve in terms of exact match. Figure 2 shows the result.

The right most (and lowest) circle corresponds to the result without restriction (i.e., threshold is 0). As threshold increased, precision gradually increased until recall fell to around 40%. Then precision sharply increased up to around 90% with recall decreasing.

Although high precision was achieved, recall steeply dropped at the same time. To prevent recall from dipping, other features need to be investigated for the certainty measure; for instance, surrounding words (contextual cues) may be effective.

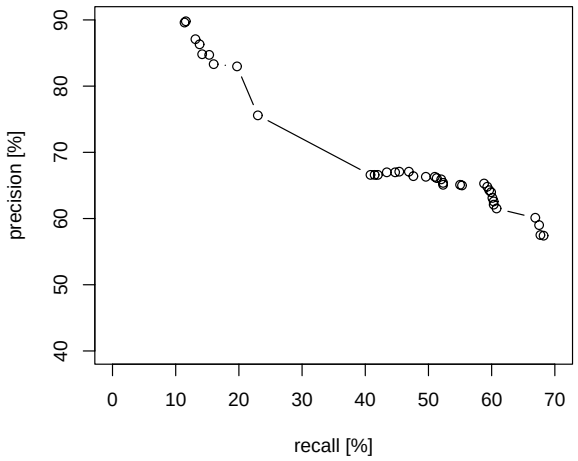

Figure 2. The relation between recall and precision for exact match.

5 Conclusions

In this paper, we presented a method for identifying protein names in biomedical texts with an emphasis on protein name boundary expansion. Our method utilizes a set of simple heuristics for initial detection of protein name fragments and takes advantage of a probabilistic model for expanding and finding protein name boundaries. The probabilistic model exploits surface clues reflecting characteristics of protein names, and combines word classes so as to avoid the data sparseness problem.

Our method, as opposed to the previous work, does not rely on POS taggers and/or syntactic parsers at all, since the information given by these NLP tools are not necessarily helpful for the task of protein name identification. This reduces both processing overhead and the potential number of probabilistic parameters to be estimated. We implemented a protein name identification system based on our proposed method, and conducted comparative experiments to verify the effectiveness of the method. The results demonstrated that our system performed well and was quite comparable to the Yapex protein name tagger which incorporates a syntactic parser. Moreover, in the case where only compound protein names were evaluated, on the whole our system outperformed Yapex, especially in exact match. Furthermore, we proposed a notion of certainty to filter out erroneous identifications for improving precision; it was demonstrated to be effective to incrementally raise precision at the expense of recall.

6 Future Work

Future work would include an incorporation of wider context in order to capture co-relations of neighboring words. Another issue to be explored is a refinement of the certainty measure; the certainty score currently applied lacks the rationale of the statistical point of view. One possible extension is to utilize the probability estimates computed for name boundary expansion. Additionally, we are planning to automatically collect large-scale training corpora in order to further improve our system performance.

Acknowledgment

We would like to thank the members of the *Proteinhalt i text* project for sharing their invaluable resources for public use. This project was partially supported by the NSF ITR grant #9817572.

References

[1] L. A. Adamic, D. Wilkinson, B. A. Huberman, and E. Adar. A literature based method for identifying gene-disease connections. In *Proceedings of the IEEE Computer Society Bioinformatics Conference (CSB 2002)*, pages 109–117, 2002.

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

[3] N. Collier, C. Nobata, and J. Tsujii. 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, 2000.

[4] B. de Bruijn and J. Martin. Literature mining in molecular biology. In *Proceedings of the Workshop on Natural Language Processing in Biomedical Applications (NLPBA 2002)*, pages 7–18, 2002.

[5] K. Franzén. Proteinhalt i text (protein concentration in text), 2003. Retrieved March 27, 2003, from `http://www.sics.se/humle/projects/prothalt/`.

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

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

[8] Y. Fu, J. Mostafa, and K. Seki. Protein association discovery in biomedical literature. In *Proceedings of the Joint Conference on Digital Libraries (JCDL 2003)*, pages 113–115, 2003.

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

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

[11] L. Hirschman, J. C. Park, J. Tsujii, L. Wong, and C. H. Wu. Accomplishments and challenges in literature data mining for biology. *Bioinformatics*, 18(12):1553–1561, 2002.

[12] J. Kazama, T. Makino, Y. Ohta, and J. Tsujii. Tuning support vector machines for biomedical named entity recognition. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain*, pages 1–8, 2002.

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

[14] M. Narayanaswamy, K. E. Ravikumar, and V. K. Shanker. A biological named entity recognizer. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 8, 2003.

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

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

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

[18] M. Palakal, M. Stephens, S. Mukhopadhyay, R. Raje, and S. Rhodes. A multi-level text mining method to extract biological relationships. In *Proceedings of the IEEE Computer Society Bioinformatics Conference (CSB 2002)*, pages 97–108, 2002.

[19] D. Proux, F. Rechenmann, and L. Julliard. 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, 1998.

[20] K. Seki and J. Mostafa. An approach to protein name extraction using heuristics and a dictionary. In *Proceedings of the American Society for Information Science and Technology Annual Conference (ASIST 2003)*, October 2003. (To appear).

[21] T. Sekimizu, H. S. Park, and J. Tsujii. 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, 1998.

[22] L. Tanabe and J. Wilbur. 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, 2002.

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

[24] I. H. Witten and T. C. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094, 1991.