Probabilistic Term Variant Generator for Biomedical Terms

Yoshimasa Tsuruoka†‡
†CREST, JST
(Japan Science and Technology Corporation)
Honcho 4-1-8, Kawaguchi-shi,
Saitama, 332-0012, Japan
tsuruoka@is.s.u-tokyo.ac.jp

Jun'ichi Tsujii‡†
‡Department of Computer Science,
University of Tokyo
Hongo 7-3-1, Bunkyo-ku,
Tokyo, 113-0033, Japan
tsujii@is.s.u-tokyo.ac.jp

ABSTRACT

This paper presents an algorithm to generate possible variants for biomedical terms. The algorithm gives each variant its generation probability representing its plausibility, which is potentially useful for query and dictionary expansions. The probabilistic rules for generating variants are automatically learned from raw texts using an existing abbreviation extraction technique. Our method, therefore, requires no linguistic knowledge or labor-intensive natural language resource. We conducted an experiment using 83,142 MEDLINE abstracts for rule induction and 18,930 abstracts for testing. The results indicate that our method will significantly increase the number of retrieved documents for long biomedical terms.

Categories and Subject Descriptors

I.2.7 [**Computing Methodologies**]: Natural Language Processing—*Language Generation*; H.3.1 [**Information Systems**]: Content Analysis and Indexing—*Thesauruses*

General Terms

Algorithms

Keywords

spelling variation, query expansion, dictionary expansion

1. INTRODUCTION

In the biomedical domain, technical terms including the names of DNA, proteins, or tissues play a central role in information retrieval. Spelling variations, however, make keyword-based approaches less effective. For example, the protein "NF-Kappa B" has many spelling variants such as "NF Kappa B," "NF kappa B," "NF kappaB," and "NFkappaB." Consequently, a large section of the documents containing this protein are not retrieved by using one of the variants as a keyword.

Dictionary-based biomedical term recognition is widely used as the first step in information extraction from biomedical documents [7, 11] because they can provide information of "id"s of recognized terms unlike machine learning based approaches [5, 6, 13]. However, one cannot achieve a high recall in recognizing long terms by using exact string matching algorithms due to the existence of spelling variations.

The problems with spelling variation in the area of information retrieval have been extensively studied in the context of query expansion and word conflation [2, 4, 8, 14, 15]. However, because the characteristics of the spelling variations for biomedical terms differ from those of common English words, we need to develop a different method for these technical terms. Manually developing rules for spelling variation, however, requires a considerable amount of labor.

In addition, a lot of technical terms consisting of many words are present in the biomedical domain. Suppose that we have a term "nuclear factor of activated T cells" and a rule that spaces and hyphens are replaceable. The number of possible variants is at least 32 because the term has five spaces. If we have another rule that the head of a word can be capitalized, the number of possible variants becomes prohibitively large. Although this problem can be alleviated by using *approximate string matching* techniques [9], such elastic matching methods require much greater computational cost than exact matching techniques.

To solve these problems, we propose a completely unsupervised method to generate spelling variants for biomedical terms. Our method not only generates spelling variants but also gives each variant a *generation probability* that represents the plausibility of the variant. Therefore, one does not receive a prohibitive number of unnecessary variants by setting the threshold of generation probability.

Our method needs no labor-intensive natural language resource, such as annotated corpora or thesauri. Our algorithm learns the rules for variant generation from raw texts. One can therefore easily adapt the generator to a new research area as soon as new documents become available. This advantage is very important because of the rapid expansion in biomedical research.

This paper is organized as follows. Section 2 describes the algorithm for generating term variants and the method in which it learns the probabilistic rules from raw texts. Section 3 presents experimental results using MEDLINE abstracts and generated variants for some biomedical terms. Section 3 also gives experimental results of dictionary-based protein name recognition using our variant generator. Section 4 offers some concluding remarks.

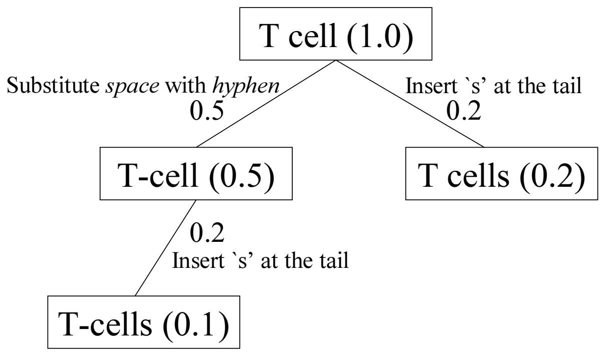

Figure 1: Probabilistic Term Variant Generation (Figures inside parentheses are generation probabilities, and figures along the edges are operation probabilities)

2. PROBABILISTIC TERM VARIANT GENERATION

2.1 Motivation

Automatic query expansion is widely known as an effective technique for enhancing retrieval performance. Suppose that one of the input terms for a retrieval system is "NF-kappa B." The term can be expanded as, "NF Kappa B" or "nf kappa b." Although the latter variant is possible, the former is clearly more plausible. Therefore, retrieval results using the former variant should be preferred. If each generated variant is associated with a numerical value representing its plausibility, the retrieval system will be able to reflect the plausibility of expanded terms when calculating the ranks of the retrieved documents.

In this paper, we quantify the plausibility of a variant with its generation probability.

2.2 Generation Probability

The *generation probability* of a variant is defined as the probability that the variant can be generated through a sequence of operations. Each operation has an *operation probability* that represents how likely it will occur. Assuming independence among the operations, the generation probability of a variant can be formalized in a recursive manner,

$$P_X = P_Y \times P_{op}, \tag{1}$$

where P_X is the generation probability of variant X, P_Y is the generation probability of variant Y from which variant X is generated, and P_{op} is the probability of the operation by which Y is transformed into X.

Figure 1 shows an example of the generation process, which can be represented as a tree. Each node represents a generated variant and its probability. Each edge represents an operation and its probability. The root node corresponds to the input term and the generation probability of the root node is 1 by definition. We can obtain the variants of an input term in order of their generation probabilities by growing a tree in a best-first manner.

2.3 Operation Probability

To calculate the generation probabilities in our formalization, we need the probability of each operation.

Table 1: Example of Obtained Rules

Left Context	Target	Right Context	Operation
c-	R	*	Replace the target with 'r'
-	R	*	Replace the target with 'r'
*	R	e	Replace the target with 'r'
*	R	el	Replace the target with 'r'
-	R	e	Replace the target with 'r'
c-	R	el	Replace the target with 'r'
*	R	*	Replace the target with 'r'

Asterisks indicate wild cards.

We used three types of operations for the generation mechanism:

- Substitution

  Replace a character with another character.
- Deletion

  Delete a character.
- Insertion

  Insert a character.

These types of operations are motivated by the ones used in *edit distance*, which is widely used for defining similarity between two strings [9]. What makes our approach unique is that we consider character-level contexts in which an operation occurs, and we estimate the probability of the operation from a large number of pairs of spelling variations.

An operation probability is defined as the probability that the operation will occur in a given context. First, we represent contexts using the neighboring characters of the operation. The following seven types of contexts are used in this paper. They differ in relative position to the target and in how much the context is specified:

- the target letter and the preceding two letters.
- the target letter and the preceding letter.
- the target letter and the following letter.
- the target letter and the following two letters.
- the target letter, the preceding letter and the following letter.
- the target letter, the preceding two letters and the following two letters.
- the target letter only.

For an operation of a substitution or a deletion, the target indicates a letter in the string. For an operation of an insertion, the target indicates a gap between two letters. For example, if the original string is "c-Rel" and the variant is "c-rel." The operation is a substitution of 'R' with 'r.' The rules obtained from this example are shown in Table 1. They correspond to the seven types of aforementioned context. The first rule indicates that if a letter 'R' is preceded by a string "c-," then one can replace the letter with 'r.'

The next step is estimating the probability of each rule. The probability should represent how likely the operation will occur in a given context.

We estimate the probabilities from a large number of pairs of spelling variants. In this paper, a pair of spelling variants is defined as follows.

- The two strings (presumably) convey the same meaning.
- The edit distance between the two strings is 1. In other words, One string must be able to transform to the other in one operation, and vice versa.

"c-Rel" and "c-rel" is an example of a pair of spelling variants. This example contains two operations, the substitution of 'R' with 'r' and the substitution of 'r' with 'R.' The way to obtain such kinds of spelling variant pairs is described later.

If we have a large set of pairs, we can estimate the operation probabilities by using the following equation.

$$P_{op} = P(operation|context) \quad (2)$$

$$\approx \frac{f(context, operation) + 1}{f(context) + 2}, \quad (3)$$

where $f(context)$ is the frequency of the occurrence of the context, and $f(context, operation)$ is the frequency of the simultaneous occurrence of the context and operation in the set of variant pairs. We adopted Laplace's smoothing (adding 1 to the numerator and 2 to the denominator).

2.4 Generation Algorithm

Once the rules and their operation probabilities are learned, we can generate variants from an input term using those rules.

The whole algorithm for variant generation is given below. Note that V represents the set of generated terms.

1. Initialization

   Add the input term to V.

2. Selection

   Select a term and an operation to be applied to it so that the generation probability of the generated term will be the highest possible.

3. Generation

   Generate a new term using the term and the operation selected in Step 2. Then, add the generated term to V.

4. Repeat

   Go back to Step 2 until the termination condition is satisfied.

In the generation step, the system applies the rule whose context matches any part of the string. If multiple rules are applicable, the rule that has the highest operation probability is used.

Because this algorithm generates variants in order of their generation probabilities, the termination condition can be that the generation probability of the generated variant is below the predefined threshold or that the number of generated variants exceeds the predefined threshold.

2.5 Acquiring Spelling Variations

To estimate the operation probabilities, we need a large number of pairs of spelling variants. If we have an annotated corpus that provides information of the semantic identity of the technical terms, we can extract the pairs of spelling variations from them. However, since constructing such corpora requires a large amount of labor and time, few such corpora are currently available.

We therefore developed a method to bring together technical terms conveying the same meaning from raw texts. Some research efforts have been done for such purposes [1, 3, 16].

In this paper, we used an abbreviation extraction technique to gather candidates of terms conveying the same meaning.

An abbreviation is a shortened form of a word or group of words. Here is an example of an abbreviation:

Alzheimer's Disease (AD)

"AD" is an abbreviation of "Alzheimer's Disease." Throughout this paper, we call the former *short form* and the latter *long form*.

The merit of using abbreviation extraction is that long forms corresponding to a same short form have relatively high possibility to have the same meaning. Table 2 shows some examples of obtained short forms and their corresponding long forms using an abbreviation extraction method.

By extracting the long form pairs with an edit distance of 1 in each set corresponding to a short form, we can obtain pairs of spelling variants such as "American"/"Americans" and "Arachidonic acid"/"arachidonic acid." As mentioned in Section 2.3, we derive probabilistic operation rules using these long form pairs.

For abbreviation extraction, we adopted a simple but effective method proposed by Schwartz and Hearst [12]. They achieved 96% precision and 82% recall on a standard test collection of biomedical documents.

The overview of their algorithm is given below.

1. Identifying short form and long form candidates

   A word inside parentheses is regarded as a short form candidate. The words preceding the parentheses are regarded as a candidate for the long form corresponding to the short form.

2. Identifying correct long forms

   Starting from the end of both the short form and the long form, move to right to left, trying to find the shortest long form that matches the short form. Every characters in the long form must be in the same order as the characters in the short form.

For further details of the algorithm, see their work [12].

3. EXPERIMENT

We applied our method to the abstracts obtained from MEDLINE, which is the largest database of biomedical literature maintained by the National Library of Medicine. The set of abstracts used for estimating the operation probabilities was obtained from the search results with the keywords (MeSH terms) "Human" AND "Cell" and the publication years 2000 and 2001. The number of abstracts was 83,142.

Table 3: Operation rules and their probabilities

Operation Probability	Left Context	Target	Right Context	Operation
0.957	*	'	*end of string*	Delete the target
0.957	*start of string*	I	m	Replace the target with 'i'
0.947	*	H	yd	Replace the target with 'h'
0.938	*	A	de	Replace the target with 'a'
0.933	*	U	r	Replace the target with 'u'
0.929	*	E	xt	Replace the target with 'e'
0.929	*	I	nd	Replace the target with 'i'
:	:	:	:	:
0.875	*	*hyphen*	*hyphen*	Delete the target
0.875	*	*hyphen*	ex	Replace the target with a *space*
0.875	*	A	dr	Replace the target with 'a'
0.875	*	A	mi	Replace the target with 'a'
:	:	:	:	:
0.750	ph		*end of string*	Insert 'y'
0.750	po		os	Insert a *hyphen*
0.750	re	*hyphen*	se	Replace the target with a *space*
0.750	rl		uk	Insert 'e'
:	:	:	:	:

An asterisk indicate a wild card

Table 2: Results of abbreviation extraction

Short form	Long form
AA	Alcoholic Anonymous American Americans Arachidonic acid arachidonic acid amino acid amino acids anaemia anemia :
AD	Alzheimer disease Alzheimer's Disease Alzheimer's disease Atopic dermatitis actinomycin D :
Abeta	amyloid beta peptide amyloid beta-peptide amyloid-beta peptide amyloid-beta peptides :
AI	Apoptotic index Apoptotic indexes Apoptotic indices apoptosis index apoptotic index apoptotic indices :
:	:

During the phase of extracting abbreviations, 5,775 short forms and 24,281 long forms were acquired. Then the operation rules were learned from 13,147 pairs of long forms.

Table 3 shows a part of the obtained rules and their probabilities. The rules totaled 14,158.

3.1 Generated Variants

The variants of some biomedical terms generated by our algorithm are shown in Tables 4 to 9. The input terms of the generator are listed in the first row of each table. The rightmost columns in each table show the frequency of a variant in another set of abstracts that was retrieved from MEDLINE with the same keywords and the publication year of 2002. These frequencies indicate how much the recall of document retrieval is improved by the variant generation.

The first two input terms "NF-kappa B" and "transcription factor" were selected from the GENIA corpus [10], which is a semantically annotated biomedical corpus consisting of 2,000 MEDLINE abstracts. Since the corpus has term boundary information, we can count the frequency of every technical term. These two terms are the two most frequent technical terms in the corpus.

Table 4 shows the results for the input term "NF-kappa B." Because the frequency of the input term is 857 and that of the variant "NF-kappaB" is 692, this variant will almost doubles the recall.

Table 5 shows the result for the input term "transcription factor." In this case, the third variant will improve the recall about 40%.

The results for the input term "antiinflammatory effect" are shown in Table 6. The results correspond to the situation where the user inputs a term of minor spelling variation. In the first variant, the word "antiinflammatory" is split into two words, "anti" and "inflammatory." Note that the frequency of the first variant is more than four times that of the input term, this variant would significantly increase the number of retrieved documents. The fifth variant "anti-inflammatory effects" is also interesting. This variant is generated by the second one. Its frequency indicates the importance of the recursive generation in our algorithm.

Table 7 shows the results for the term "type I." The sixth

Table 4: Generated term variants for "NF-kappa B"

Generation Probability	Generated String	Frequency
1.0 (input term)	NF-kappa B	857
0.417	NF-kappaB	692
0.417	nF-kappa B	0
0.337	Nf-kappa B	0
0.275	NF kappa B	25
0.226	NF-kappa b	0
0.177	NF-kppa B	0
0.174	nF-kappaB	0
0.168	Nf kappa B	0
0.168	Nfkappa B	0
0.140	nf-kappa B	0
:	:	:

Table 5: Generated term variants for "transcription factor"

Generation Probability	Generated String	Frequency
1.0 (input term)	transcription factor	1020
0.235	Transcription factor	13
0.131	transcription factors	377
0.119	transcription-factor	0
0.090	trancription factor	0
0.031	Transcription factors	9
0.029	transcriPtion factor	0
0.028	Transcription-factor	0
0.028	transcription Factor	0
0.023	transcriptionfactor	0
0.021	Trancription factor	0
:	:	

Table 6: Generated term variants for "antiinflammatory effect"

Generation Probability	Generated String	Frequency
1.0 (input term)	antiinflammatory effect	7
0.462	anti-inflammatory effect	33
0.393	antiinflammatory effects	6
0.356	Antiinflammatory effect	0
0.286	antiinflammatory-effect	0
0.181	anti-inflammatory effects	23
0.164	Anti-inflammatory effect	3
0.140	Antiinflammatory effects	0
0.132	anti-inflammatory-effect	0
0.127	anti inflammatory effect	0
0.112	antiinflammatory-effects	0
:	:	:

Table 7: Generated term variants for "type I"

Generation Probability	Generated String	Frequency
1.0 (input term)	type I	678
0.300	Type I	1082
0.299	type i	0
0.108	type-I	11
0.090	Type i	0
0.060	type Is	0
0.046	type 1	916
0.042	tyPe I	0
0.037	type is	47
0.032	Type-I	0
0.032	type-i	1
:	:	:

variant "type 1" is interesting, where the character "I" is replaced with a figure "1." However, the results include a harmful one. The eighth variant "type is" is clearly wrong and its frequency indicate that it will cause false positives.

The results for the input term "tumour necrosis factor alpha" are shown in Table 8. Notice that the transformation to the American spelling variation of "tumour" is ranked at the top. Because the input term includes a minor spelling variation in this case, the generated variants significantly improve the recall of the document retrieval.

The results for the input term "activated T cell" are shown in Table 9. Notice that hyphens are inserted between 'T' and "cell" while they are not inserted between "activated" and 'T.' In this case, the frequency of the input term and the sum of the frequencies of the variants are almost equal. This suggests a potential doubling of the retrieved documents.

3.2 Dictionary Expansion

Applications of our algorithm include dictionary expansion, where each entry in a dictionary is expanded by a variant generator.

As an example of dictionary expansion, we conducted an experiment of dictionary-based gene/protein name recognition using the GENIA corpus.

The dictionary was constructed from GenBank, which is one of the largest gene databases containing 476,296 names of genes and proteins.[1] The names were filtered excluding frequently occurring English words and entries consisting only of numbers.

Expansions were done on terms whose length was equal to or longer than five characters. The threshold of the generation probability was set to 0.1. The maximum number of variants generated for each term was limited to 20.

Because we were not concerned with the absolute recognition performance in this paper, we simply used the longest match algorithm for name recognition and no other techniques to boost recognition performance.

Each recognition was counted correct if both the boundaries of the recognized term exactly matched the boundaries of a semantic annotation in the corpus. The semantic annotations used in this experiments were of DNA, RNA, proteins, and their descendants in the GENIA ontology [10]. When a term was recursively annotated, only the innermost (shortest) annotation was considered.

The results are shown in Figures 2 and 3. The dotted lines are the recognition performance using the original dictionary, and the solid lines are the performance using the expanded dictionary. The precisions and recalls were averaged over five consecutive lengths.

Notice that the recalls for medium length terms (consisting of from 6 to 10 letters) were almost doubled by the dictionary expansion. The recalls for long terms (consisting of

[1]We used the gene symbol index file obtained from ftp://ncbi.nlm.nih.gov/genbank/gbgen.idx.Z

Table 8: Generated term variants for "tumour necrosis factor alpha"

Generation Probability	Generated String	Frequency
1.0 (input term)	tumour necrosis factor alpha	15
0.492	tumor necrosis factor alpha	126
0.356	tumour necrosis factor-alpha	30
0.235	Tumour necrosis factor alpha	2
0.175	tumor necrosis factor-alpha	182
0.115	Tumor necrosis factor alpha	8
0.102	tumour-necrosis factor alpha	0
0.099	tumour necrosis factoralpha	0
0.084	Tumour necrosis factor-alpha	4
0.075	tumour necrosis-factor alpha	0
0.075	tumor-necrosis factor alpha	0
:	:	:

Table 9: Generated term variants for "activated T cell"

Generation Probability	Generated String	Frequency
1.0 (input term)	activated T cell	99
0.639	activated T cells	90
0.475	activated T-cell	10
0.469	Activated T cell	1
0.304	activated T-cells	5
0.299	Activated T cells	1
0.223	Activated T-cell	0
0.203	activated t cell	0
0.142	Activated T-cells	0
0.130	activated t cells	0
0.097	activated t-cell	0
:	:	:

more than 10 letters) were also improved. The fact that the precisions were slightly improved indicates that our algorithm generates few variants that cause false recognition.

4. CONCLUSION

An algorithm for generating possible variants for a given biomedical term was presented. This algorithm can be used for many applications regarding information extraction from biomedical documents, such as query expansion or dictionary expansion.

The learning process is done in a completely unsupervised way. A large number of pairs of spelling variations are retrieved from raw texts using an existing abbreviation extraction technique. The probabilistic rules for generating the variants are learned from those pairs.

The experimental results of variant generation for some long biomedical terms indicate that our method will significantly improve the recall of document retrieval without the burden of generating a large number of unnecessary variants.

The experimental results of the dictionary expansion using a gene/protein name dictionary and the GENIA corpus show that our algorithm considerably improves the recall for medium length biomedical terms without any loss of precision.

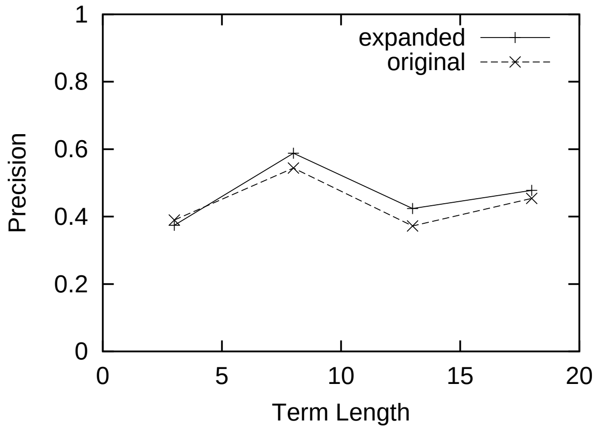

Figure 2: Precision in dictionary-based protein name recognition

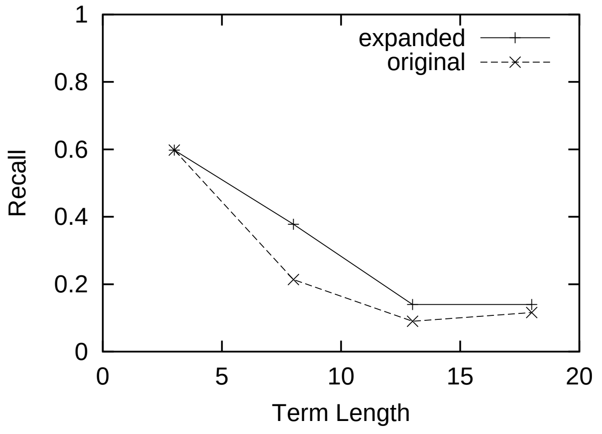

Figure 3: Recall in dictionary-based protein name recognition

4.1 Future work

Three types of operations are considered in this paper for the mechanism of variant generation. There can be, however, other types of operations, such as word-insertion and word-replacement. Our future work should encompass those types of operations to improve the recall for long biomedical terms.

We have applied our method to biomedical documents. However, our method requires only that the technical terms in the domain often be abbreviated so that one can collect a large number of pairs of spelling variants by extracting abbreviations. Studying the applicability of our method to other domains is one of the future directions in our research.

5. REFERENCES

[1] M. Baroni, J. Matiasek, and H. Trost. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 48–57, 2002.

[2] D. A. Hull. Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society of Information Science*, 47(1):70–84, 1996.

[3] C. Jacquemin. Guessing morphology from terms and corpora. In *Research and Development in Information Retrieval*, pages 156–165, 1997.

[4] C. Jacquemin and E. Tzoukermann. *NLP for term variant extraction: Synergy between Morphology, Lexicon and Syntax*, pages 25–74. Kluwer Academic Publishers, 1999.

[5] J. Kazama, T. Makino, Y. Ohta, and J. Tsujii. Tuning support vector machines for biomedical named entity recognition. In *Proceedings of the ACL-02 Workshop on Natural Language Processing in the Biomedical Domain*, 2002.

[6] J. D. Kim and J. Tsujii. Corpus-based approach to biological entity recognition. In *Proceedings of the Second Meeting of the Special Interest Group on Text Data Mining of ISMB 2002*, 2002.

[7] M. Krauthammer, A. Rzhetsky, P. Morozov, and C. Friedman. Using blast for identifying gene and protein names in journal articles. *GENE*, 259:245–252, 2000.

[8] R. Krovetz. Viewing Morphology as an Inference Process,. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–203, 1993.

[9] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[10] T. Ohta, Y. Tateisi, J.-D. Kim, and J. Tsujii. Genia corpus: an annotated research abstract corpus in molecular biology domain. In *Proceedings of the Human Language Technology Conference (HLT 2002)*, 2002.

[11] T. Ono, H. Hishigaki, A. Tanigami, and T. Takagi. Automated extraction of information on protein-protein interactions from the biological literature. *BIOINFORMATICS*, 17(2):155–161, 2001.

[12] A. Schwartz and M. Hearst. A simple algorithm for identifying abbreviation definitions in biomedical texts,. In *Proceedings of the Pacific Symposium on Biocomputing (PSB 2003)*, 2003.

[13] K. Takeuchi and N. Collier. Use of support vector machines in extended named entity recognition. In *Proceedings of the sixth Conference on Natural Language Learning (CoNLL-2002), Roth, D. and van den Bosch, A. (eds)*, pages 119–125, 2002.

[14] E. Tzoukermann, J. Klavans, and C. Jacquemin. Effective use of natural language processing techniques for automatic conflation of multi-word terms: The role of derivational morphology, part of speech tagging, and shallow parsing. In *Research and Development in Information Retrieval*, pages 148–155, 1997.

[15] J. Xu and W. B. Croft. Corpus-based stemming using cooccurrence of word variants. *ACM Transactions on Information Systems*, 16(1):61–81, 1998.

[16] P. Zweigenbaum and N. Grabar. Automatic acquisition of morphological knowledge for medical language processing. *Lecture Notes in Computer Science*, 1620:416–420, 1999.