International Journal of Medical Informatics (2006) **75**, 456—467

ELSEVIER

International Journal of Medical Informatics

www.intl.elsevierhealth.com/journals/ijmi

Recognizing names in biomedical texts using mutual information independence model and SVM plus sigmoid

G.D. Zhou*

Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613, Singapore

Received 15 April 2005; received in revised form 30 June 2005; accepted 30 June 2005

KEYWORDS
Biomedical name recognition;
Mutual information independence model;
Support vector machine

Summary In this paper, we present a biomedical name recognition system, called PowerBioNE. In order to deal with the special phenomena in the biomedical domain, various evidential features are proposed and integrated through a mutual information independence model (MIIM). In addition, a support vector machine (SVM) plus sigmoid is proposed to resolve the data sparseness problem in the MIIM. In this way, the data sparseness problem in MIIM-based biomedical name recognition can be resolved effectively and a biomedical name recognition system with better performance and better portability can be achieved. Finally, we present two post-processing modules to deal with the nested entity name and abbreviation phenomena in the biomedical domain to further improve the performance. Evaluation shows that our system achieves *F*-measures of 69.1 and 71.2 on the 23 classes of GENIA V1.1 and V3.0, respectively. In particular, our system achieves an *F*-measure of 77.8 on the ''*protein*'' class of GENIA V3.0. It also shows that our system outperforms the best-reported system on GENIA V1.1 and V3.0.

1. Introduction

With an overwhelming amount of textual information in molecular biology and biomedicine, there is a need for effective and efficient literature mining and knowledge discovery that can help biologists to gather and make use of the knowledge encoded in text documents. In order to make organized and structured information available, automatically recognizing biomedical entity names becomes critical and is important for protein—protein interaction extraction, pathway construction, automatic database duration, etc.

Such a task, called named entity recognition, has been well developed in the newswire domain [1—4]. In MUC and CoNLL, the task of named entity recognition is to recognize the names of persons, locations, organizations, etc. in the newswire domain. Representative machine learning approaches include hidden Markov model

* Tel.: +65 68745055.
E-mail address: zhougd@i2r.a-star.edu.sg.

1386-5056/$ — see front matter © 2005 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.ijmedinf.2005.06.012

[5–8], maximum entropy [9–12], conditional random fields [13], support vector machine [14], and decision tree [15]. In the biomedical domain, we care about entities like gene, protein, virus, etc. In recent years, many biomedical name recognition systems [16–22,14,23–25] have been developed or ported from existing named entity recognition systems in the newswire domain. However, few of them have achieved satisfactory performance due to the special characteristics in the biomedical domain, such as long and descriptive naming convention, the conjunctive and disjunctive structures, casual naming convention and the rapid emergence of new biomedical names, abbreviation, and nested construction. On all accounts, we can say that recognition of entity names in the biomedical domain is much more difficult than that in the newswire domain.

This paper will follow hidden Markov model (HMM) due to: (1) its great success in many tagging applications, most notably part-of-speech (POS) tagging [26–28] and shallow/full parsing [29,30], besides named entity recognition; (2) the efficiency of the Viterbi algorithm [31] in global optimization during decoding the state sequence. In theory, HMM is based on a joint probability model. Given an observation sequence $O_1^n = o_1 o_2 \cdots o_n$ in sequential inference, the goal of a joint probability model to find a stochastic optimal state sequence $S_1^n = s_1 s_2 \cdots s_n$ that maximizes the joint probability $P(S_1^n, O_1^n)$

$$\begin{aligned} S^* &= \arg\max_{S_1^n} \log P(S_1^n, O_1^n) \\ &= \arg\max_{S_1^n} \{\log P(S_1^n) + \log P(O_1^n|S_1^n)\} \end{aligned} \quad (1)$$

Traditionally, a HMM segments and labels sequential data in a generative way by making an observation context independent assumption that successive observations are independent given the corresponding individual state [32]

$$P(O_1^n|S_1^n) = \prod_{i=1}^{n} P(o_i|s_i) \quad (2)$$

By applying the above assumption and using the chain rule, a HMM can be expressed as Eq. (3) by rewriting Eq. (1)

$$S^* = \arg\max_{S_1^n} \left\{ \log P(S_1^n) + \sum_{i=1}^{n} \log P(o_i|s_i) \right\} = \arg\max_{S_1^n} \left\{ \sum_{i=2}^{n} \log P(s_i|S_1^{i-1}) + \log P(s_1) + \sum_{i=1}^{n} \log P(o_i|s_i) \right\} \quad (3)$$

The above HMM consists of two models: the state transition model $\sum_{i=2}^{n} \log P(s_i|S_1^{i-1}) + \log P(s_1)$ as the first two terms in Eq. (3) and the output model $\sum_{i=1}^{n} \log P(o_i|s_i)$ as the third term in Eq. (3).

There are several problems with this generative approach. First, many tasks would benefit from a richer representation of observations—in particular a representation that describes observations in terms of many overlapping features, such as capitalization, word ending, part-of-speech in addition to the traditional word identity. Note that these features always depend on each other. Second, while the dependence between successive states can be directly modeled by its state transition model, the generative approach fails to effectively model the dependence in the observation sequence due to its strong context independent assumption. Third, in many NLP tasks, the goal is to predict the state sequence given the observation sequence. In other words, the generative approach inappropriately applies a generative joint probability model for a conditional probability problem. In summary, the main reasons behind these problems of the generative approach are the strong context independent assumption and the generative nature in modeling sequential data.

To resolve above problems, this paper presents the mutual information independence model (MIIM), a discriminative Markov model, which models a conditional probability in a discriminative way. Besides, various evidential features are proposed to deal with the special phenomena in the biomedical domain and integrated effectively and efficiently through the MIIM. In addition, a support vector machine (SVM) plus sigmoid is proposed to resolve the data sparseness problem in our system. Finally, we present two post-processing modules to deal with the nested entity name and abbreviation phenomena to further improve the performance.

All of our experiments are done on the GENIA corpus, which is the largest annotated corpus in the molecular biology domain available to public [33]. In our experiments, two versions are used: (1) GENIA V1.1, which contains 670 MEDLINE abstracts of 123K words; (2) GENIA V3.0 which is a superset of GENIA V1.1 and contains 2000 MEDLINE abstracts of 360K words. The annotation of biomedical entities is based on the GENIA ontology [33], which

includes 23 distinct classes: multi-cell, mono-cell, virus, body part, tissue, cell type, cell component, organism, cell line, other artificial source, protein, peptide, amino acid monomer, DNA, RNA, poly nucleotide, nucleotide, lipid, carbohydrate, other organic compound, inorganic, atom and other.

The layout of this paper is as follows: In Section 2, we will introduce various features to deal with the special phenomena of entity naming conventions in the biomedical domain. In Section 3, we will present the mutual information independence model (MIIM) to integrate various features, a support vector machine (SVM) plus sigmoid to resolve the data sparseness problem in the MIIM and two post-processing modules to further improve the performance. In Section 4, we will evaluate the detailed performance on the GENIA corpus while a detailed error analysis is presented in Section 5. Related work in biomedical name recognition is introduced in Section 6 while some conclusion remarks are given in Section 7.

2. Features

In order to deal with the special phenomena of entity naming conventions in the biomedical domain, such as long and descriptive naming convention, casual naming convention, new biomedical names rapidly emerging and abbreviation [24,25], various evidential features are explored in this paper.

- *Word Formation Pattern* (F_{WFP}): The purpose of this feature is to capture capitalization, digitalization and other word formation information. This feature has been widely used in both the newswire domain [6,11,7] and the biomedical domain [18,21,20,14,22—25]. Like Shen et al. [24], Table 1 shows the list of word formation patterns in our system with descending order of priority.
- *Morphological Pattern* (F_{MP}): Morphological information, such as prefix and suffix, is considered as an important cue for terminology identification and has been widely applied in the biomedical domain [22—25].

Like Shen et al. [24], a statistical method is first applied to get the most frequent prefixes/suffixes from the training data as candidates. Then, each of these candidates is evaluated using

$$W_i = \frac{\#\mathrm{IN}_i - \#\mathrm{OUT}_i}{\#\mathrm{IN}_i + \#\mathrm{OUT}_i} \quad (4)$$

Table 1 F_{WFP}: list of word formation patterns in descending order of priority

F_{WFP}	Example
Comma	,
Dot	.
LRB	(
RRB	)
LSB	[
RSB	]
RomanDigit	II
GreekLetter	Beta
StopWord	In, at
ATCGsequence	AACAAAG
OneDigit	5
AllDigits	60
DigitCommaDigit	1.25
DigitDotDigit	0.5
OneCap	T
AllCaps	CSF
CapLow Alpha	All
CapMixAlpha	IgM
LowMixAlpha	kDa
AlphaDigitAlpha	H2A
AlphaDigit	T4
DigitAlphaDigit	6C2
DigitAlpha	19D

where $\#\mathrm{IN}_i$ is the number of the ith candidate occurring within entity names and $\#\mathrm{OUT}_i$ is the number of the ith candidate occurring outside entity names.

The rationale behind this method is that a particular prefix/suffix, which occurs most likely within entity names, may be thought of as evidence for distinguishing entity names. The candidates whose weights are above a certain threshold are chosen. Then, we count the frequency of each prefix/suffix in each entity class and group prefixes/suffixes with the similar distribution among the entity classes into one category. Each category is labeled by its most occurring entity class. Therefore, each prefix/suffix ends up being associated with exactly one of the entity classes. This can help resolve the data sparseness problem because prefixes/suffixes with similar distribution have similar contribution. Here, 0.7 is chosen as the threshold for the candidate weight (as shown in Eq. (4)) based on our experimentation. As a result, average 37 prefixes/suffixes are selected from the training data of GENIA V3.0 and further grouped to 23 categories related with the 23 classes in the GENIA ontology.

Table 2 shows some of morphological patterns. It shows that suffixes ~*Lipid*, ~*Rogen*, ~*Vitamin* have been grouped into the category $\mathrm{MP}_{\mathrm{LIPID}}$ because they occur frequently in the class ''*Lipid*'' and much less frequently in other classes.

Table 2 F_{WFP}: examples of morphological patterns

F_{MP}	Prefix/suffix	Example
MP_{LIPID}	~Lipid	Phospholipids
	~Rogen	Estrogen
	~Vitamin	Dihydroxyvitamin
MP_{VIRUS}	~Virus	Cytomegalovirus

- *Part-of-speech* (F_{POS}): Since many of the words in biomedical names are in lowercase, capitalization information in the biomedical domain is not as evidential as that in the newswire domain. Moreover, many biomedical names are descriptive and very long. Therefore, POS may provide useful evidence about the boundaries of biomedical entity names [23–25].
- *Head noun trigger* (F_{HEAD}): The head noun, which is the major noun of a noun phrase, often describes the function or the property of the noun phrase. Like Shen et al. [24], a list of head noun triggers is extracted as follow: First, we automatically extract unigram and bigram head nouns from the training data, and rank them by frequency. Then, for each entity class, we select 50% of top ranked head nouns as head noun triggers. Table 3 shows some of the examples.
- *Name alias feature* (F_{ALIAS}): Besides the above widely used features, a novel name alias feature is also proposed to resolve the name alias phenomenon. The intuition behind this feature is the name alias phenomenon that relevant entities will be referred to in many ways throughout a given text and thus success of named entity recognition is conditional on success at determining when one noun phrase refers to the very same entity as another noun phrase.

During decoding, the entity names already recognized from the previous sentences of the document are stored in a list. When the system encounters an entity name candidate (e.g. a word with a special word formation pattern), a name alias algorithm (similar to [34]) is invoked to first dynamically determine whether the entity name candidate might be alias for a previously recognized name in the recognized list. This is done by checking whether all the characters in the entity name candidate exist in a recognized entity name in the same order and whether the first character in the entity name candidate is same as the first character in the recognized name. For a relevant work, please see [35]. The name alias feature F_{ALIAS} is represented as *ENTITYnLm* (L indicates the locality of the name alias phenomenon). Here *ENTITY* indicates the class of the recognized entity name and *n* indicates the number of words in the recognized entity name while *m* indicates the number of words in the recognized entity name from which the name alias candidate is formed. For example, when the decoding process encounters the word ''*TCF*'', the word ''*TCF*'' is proposed as an entity name candidate and the name alias algorithm is invoked to check if the word ''*TCF*'' is an alias of a recognized named entity. If ''*T cell factor*'' is a ''*Protein*'' name recognized earlier in the document, the word ''*TCF*'' is determined to be an alias of ''*T cell factor*'' with the name alias feature *Protein3L3* by taking the three initial letters of the three-word ''*protein*'' name ''*T cell factor*''.

Table 3 F_{HEAD}: examples of auto-generated head noun triggers

Class	Unigram	Bigram
Protein	Interleukin	Activator protein
	Interferon	Binding protein
	Kinase	Cell receptor
DNA	DNA	X chromosome
	cDNA	Binding motif
	Chromosome	Promoter element

3. Methods

3.1. Mutual information independence model

Given above various features, the key problem is how to effectively and efficiently integrate them together and find the optimal resolution to biomedical name recognition. More formally in conditional probability, given an observation sequence $O_1^n = o_1 o_2 \cdots o_n$, the purpose is to find the most likely biomedical name tag sequence $S_1^n = s_1 s_2 \cdots s_n$ that maximizes $P(S_1^n|O_1^n)$. Here, the observation $o_i = \langle f_i, w_i \rangle$, w_i is the ith word, $f_i = \langle F^i_{WFP}, F^i_{MP}, F^i_{POS}, F^i_{HEAD}, F^i_{ALIAS} \rangle$ is the feature set of the word w_i, and the biomedical name tag s_i is a structural state $s_i = BOUNDARY_i\_ENTITY_i\_FEATURE_i$, where

- $BOUNDARY_i$ {O, B, M, E} denotes the position of the current word in the entity. Here O means that current word is a whOle entity name and B/M/E means that current word is at the Beginning/in the Middle/at the End of an entity name.
- $ENTITY_i$ indicates the class of the entity.
- $FEATURE_i$ is the word formation pattern feature F^i_{WFP}. Because of the limited number of bound-

ary and entity categories, the word formation pattern feature is added into the structural tag to represent a more accurate state transition model in the MIIM while keeping the output model unchanged. That is, the structural tag is factored by the word formation pattern feature to achieve a more detailed and powerful state transition model in the MIIM.

In this paper, we use the mutual information independence model (MIIM), a discriminative model as described in Zhou and Su [7]. Similar to hidden Markov model (HMM), a MIIM is a latent variable model in the sense that only the observation sequence is given while the state (tag) sequence remains ''hidden''. In principle, given an observation sequence $O_1^n = o_1 o_2 \cdots o_n$, the goal of a conditional probability model is to find a stochastic optimal state sequence $S_1^n = s_1 s_2 \cdots s_n$ that maximizes the conditional probability $P(S_1^n|O_1^n)$

$$S^* = \arg\max_{S_1^n} \{\log P(S_1^n|O_1^n)\} \tag{5}$$

By rewriting $\log P(S_1^n|O_1^n)$, we have

$$\begin{aligned} S^* &= \arg\max_{S_1^n} \{\log P(S_1^n|O_1^n)\} \\ &= \arg\max_{S_1^n} \left\{ \log P(S_1^n) + \log \frac{P(S_1^n, O_1^n)}{P(S_1^n) \cdot P(O_1^n)} \right\} \end{aligned} \tag{6}$$

Obviously, the second term in Eq. (6) is the pair-wise mutual information between S_1^n and O_1^n, which means the change of information content when S_1^n and O_1^n co-occur. In order to simplify the computation of this term, we assume a novel pair-wise mutual information independence

$$\begin{aligned} &\mathrm{MI}(S_1^n, O_1^n) = \sum_{i=1}^{n} \mathrm{MI}(s_i, O_1^n) \quad \text{or} \\ &\log \frac{P(S_1^n, O_1^n)}{P(S_1^n) \cdot P(O_1^n)} = \sum_{i=1}^{n} \log \frac{P(s_i, O_1^n)}{P(s_i) \cdot P(O_1^n)} \end{aligned} \tag{7}$$

This assumes that the reduction in uncertainty for the state sequence in knowing the observation sequence is equal to the summation of the reduction in uncertainty for each individual state of the state sequence in knowing the observation sequence. That is, an individual tag is only dependent on the observation sequence O_1^n and independent of other tags in the tag sequence S_1^n. This assumption is reasonable because the dependence among the tags in the tag sequence S_1^n has already been captured by the first term in Eq. (6). Applying the Eq. (7) into Eq. (6), we have Eq. (8) as follows:

$$\begin{aligned} S^* &= \arg\max_{S_1^n} \log P(S_1^n) + \sum_{i=1}^{n} \log \frac{P(s_i, O_1^n)}{P(s_i) \cdot P(O_1^n)} = \arg\max_{S_1^n} \left\{ \log P(S_1^n) - \sum_{i=1}^{n} \log P(s_i) + \sum_{i=1}^{n} \log \frac{P(s_i, O_1^n)}{P(O_1^n)} \right\} \\ &= \arg\max_{S_1^n} \left\{ \sum_{i=2}^{n} \log P(s_i|S_1^n) + \log P(s_1) - \sum_{i=1}^{n} \log P(s_i) + \sum_{i=1}^{n} \log P(s_i|O_1^n) \right\} \\ &= \arg\max_{S_1^n} \left\{ \sum_{i=2}^{n} \log P(s_i|S_1^{i-1}) - \sum_{i=2}^{n} \log P(s_i) + \sum_{i=1}^{n} \log P(s_i|O_1^n) \right\} \end{aligned} \tag{8}$$

The above model consists of two models: the state transition model $\sum_{i=2}^{n} \log P(s_i|S_1^{i-1}) - \sum_{i=2}^{n} \log P(s_i)$ as the first two terms in Eq. (8) and the output model $\sum_{i=1}^{n} \log P(s_i|O_1^{i-1})$ as the third term in Eq. (8). We call the above model as shown in Eq. (8) the mutual information independence model due to its pairwise mutual information assumption as shown in Eq. (7). Compared with the generative HMM as shown in Eq. (3), the main difference lies in their output models in that the output model of a MIIM directly captures the context dependence between successive observations in determining the ''hidden'' states while the output model of the generative HMM fails to do so. That is, the output model of a MIIM overcomes the strong context independent assumption in the generative HMM and becomes observation context dependent. On the one hand, the state transition model of the MIIM takes advantage of a generative HMM on modeling sequential states. On the other hand, the output model of the MIIM also takes advantage of a discriminative model on incorporating arbitrary overlapping features. Moreover, the output model of the MIIM directly captures the context dependence between successive observations in determining the ''hidden'' states individually and extending the context dependence to the entire observation sequence.

The idea behind the MIIM is that it tries to assign each observation an appropriate tag (state), which contains boundary and class information. For example, ''*TCF 1* binds stronger than *NF kB* to *TCEd DNA*''. The tag assigned to token ''*TCF*'' should

indicate that it is at the beginning of an entity name and it belongs to the ''*Protein*'' class; and the tag assigned to token ''*binds*'' should indicate that it does not belong to an entity name. Here, a variant of the Viterbi algorithm [31] in decoding the standard HMM is implemented to find the most likely tag sequence by replacing the state transition model and the output model of the standard HMM with the state transition model and the output model of the MIIM, respectively.

From Eq. (8), we can see that the first term can be computed by using ngram modeling [36], where each tag is assumed to be dependent on the $N-1$ previous tags (e.g. 2), and that the second term can be easily calculated by summing log probabilities of all the individual tags.

The problem with the above MIIM lies in the data sparseness problem raised by its output model: $P(s_i|O_1^n)$ in the third term of Eq. (8). Ideally, we would have sufficient training data for every event whose conditional probability we wish to calculate. Unfortunately, there is rarely enough training data to compute accurate probabilities when decoding on new data. Generally, two smoothing approaches [36] are applied to resolve this problem: linear interpolation and back-off. However, these two approaches only work well when the number of different information sources is limited. When a few features and/or a long context are considered, the number of different information sources is exponential. This makes smoothing approaches inappropriate in our system. Therefore, a support vector machine (SVM) plus sigmoid is proposed to resolve the data sparseness problem in our system. The reason that we choose SVM for this purpose is that it represents the state-of-the-art in the machine learning research community and there are good implementations of the algorithm available. In this paper, we use the SVMLight[1] deleveloped by Joachims [37].

3.2. Support vector machine plus sigmoid

Support vector machines (SVMs) are a popular machine learning approach first presented by Vapnik [38]. Based on the structural risk minimization of the statistical learning theory, SVMs seek an optimal separating hyper-plane to divide the training instances into two classes and the decisions on the testing instances are made based on the support vectors which are selected as the only effective instances in the training set.

Normally, a window of a target word w represents the local context of w and is used to make a decision on w. In our system, we set the window size to 7, which includes the previous three words and the next three words of the target word w including the target word w itself. Here, each instance in the training and test data is represented using a high-dimensional feature vector. For example, if a word occurs in the vocabulary (collected from the training data), one dimension in the feature vector of the SVM (corresponding to the position of the word in the vocabulary) is set to 1. The vocabulary is constructed by taking all the words in the training data (filtered with threshold 3). In our system, all the five features as described in Section 2 are applied for each of the seven words in the window. When a word contains dash(es), one additional overlapping word formation pattern feature is generated for each segment separated by dashes. For example, the word ''TCF-Beta'' has not only a word formation pattern feature of ''CapMixAlpha'' as a whole but also two additional overlapping word formation pattern features ''AllCaps'' and ''GreekLetter'' for the two segments separated by the dash while the word ''TCF'' only has a word formation pattern feature of ''AllCaps''.

However, in order to apply SVM in the output model of the MIIM (the third term of Eq. (4)), we need to solve three problems.

First, SVMs only produce an un-calibrated value that is not a probability. The unthresholded output of an SVM can be represented as

$$f(x) = \sum_{i \in SV} a_i \cdot y_i \cdot k(x_i, x) + b \tag{9}$$

while x is the given example to be classified; b is the offset in the SVM classifier; x_i, y_i and a_i are a support vector, the output of the support vector and its weight, repectively; $k(x, x_i)$ is the kernel (similarity) function between the given example x and a support vector x_i. Therefore, we need to map the SVM output into a pseudo-probability. For this purpose, we train an additional sigmoid model [39] which has two parameters A and B

$$p(s_i|f_i) = \frac{1}{1+\exp(Af_i + B)} \tag{10}$$

Second, SVMs are binary classifiers. Therefore, we must extend SVMs to multi-class (e.g., K) classifiers. For efficiency, we apply the *one* versus *others* strategy, which builds K classifiers so as to separate one class from all others, instead of the *pairwise* strategy, which builds $K^*(K-1)/2$ classifiers considering all pairs of classes. Moreover, we only apply the simple linear kernel due to speed and memory requirements, although other kernels (e.g. polyno-

[1] http://svmlight.joachims.org/.

mial kernel) and the pairwise strategy can have better performance.

Finally, for each state s_i, there is one sigmoid output $p(s_i|f_i)$. Therefore, the sigmoid outputs need to be normalized to get a valid probability distribution. This can be simply done by applying

$$p(s_i|O_1^n) = \frac{p(s_i|f_i)}{\sum_i p(s_i|f_i)} \quad (11)$$

3.3. Post-processing

Two post-processing modules, namely nested entity name resolution and abbreviation resolution, are applied in our system to further improve the performance.

3.3.1. Nested entity name resolution

It is found [24] that 16.57% of entity names in GENIA V3.0 have nested constructions, e.g.

< RNA >< DNA > CIITA < /DNA > mRNA < /RNA >

Therefore, it is important to resolve such phenomenon.

Here, a pattern-based approach is proposed to resolve the nested entity names while the above MIIM and SVM plus sigmoid is applied to recognize embedded entity names and non-nested entity names. The main idea is that we try to develop a set of patterns, which help recognize biomedical names to the longest extent based on embedded ones. In the GENIA corpus, we find that there are six useful patterns of nested entity name constructions:

- <ENTITY> := <ENTITY> [head noun], e.g. <DNA> := <PROTEIN> binding motif
- <ENTITY> := <ENTITY> <ENTITY>, e.g. <PROTEIN> := <LIPID> <PROTEIN>
- <ENTITY> := [modifier] <ENTITY>, e.g. <Protein> := anti <Protein>
- <ENTITY> := <ENTITY> [any one word] <ENTITY>, e.g. <MULTICELL> := <VIRUS> infected <MULTICELL>
- <ENTITY> := <ENTITY> <ENTITY> [head noun]
- <ENTITY> := [modifier] <ENTITY> [head noun]

In our system, the above six patterns and respective rules are generated as follows: Firstly, the first four basic patterns are generated based on the manual investigation of the nested annotation in the corpus. Then, we replace the nested entity names by their entity class. For example, ''<DNA> <VIRUS>HIV-1</VIRUS> enhancer </DNA>'' has the pattern of ''<ENTITY> := <ENTITY> [head noun]''. We can replace the entity name ''HIV-1'' by ''<VIRUS>'' and transform the nested annotation to a rule ''<DNA> := <VIRUS> enhancer''. In this way, we can automatically collect all these nested rules according to the defined patterns from the training data and rank them by frequency. In our system, only the rules that have frequency not less than 2 are kept. Finally, in order to broaden the rule coverage, two additional patterns (i.e. the last two patterns as shown above) are generated by combining the first two patterns and the next two patterns, respectively. As a result, more rules are generated according to the two additional patterns by combining the existing rules generated on the first four basic patterns.

3.3.2. Abbreviation resolution

While the name alias feature is useful to detect the inter-sentential name alias phenomenon (name alias and its full form appear in different sentences), it is unable to identify the intra-sentential name alias phenomenon: the intra-sentential abbreviation (abbreviation and its full form appear in the same sentence). Such abbreviations widely occur in the biomedical domain.

In our system, we present an effective and efficient algorithm to recognize the intra-sentential abbreviations more accurately by mapping them to their full expanded forms. In the GENIA corpus, we observe that the expanded form and its abbreviation often occur together via parentheses. Generally, there are two patterns: ''expanded form (abbreviation)'' and ''abbreviation (expanded form)''.

Our algorithm is based on the fact that it is much harder to classify an abbreviation than its expanded form. Generally, the expanded form is more evidential than its abbreviation to determine its class. The algorithm works as follows: Given a sentence with parentheses, we use a similar algorithm as in Schwartz and Hearst [34] to determine whether it is an abbreviation with parentheses. This is done by starting from the end of both the abbreviation and the expanded form, moving from right to left and trying to find the shortest expanded form that matches the abbreviation. Any character in the expanded form can match a character in the abbreviation with one exception: the match of the character at the beginning of the abbreviation must match the first alphabetic character of the first word in the expanded form. If yes, we remove the abbreviation and the parentheses from the sentence. After the sentence is processed, we restore the abbreviation with parentheses to its original position in the sentence. Then, the abbreviation is classified as the same class of the expanded form, if the expanded form is recognized as an entity name.

Table 4 Performance of our PowerBioNE system (five-fold cross-validation)

Performance	*P*	*R*	*F*
Shen et al. on GENIAV3.0	66.5	66.6	66.6
Shen et al. on GENIAV1.1	63.1	61.2	62.2
Our system on GENIA V3.0	72.7	69.8	71.2
Our system on GENIA V1. 1	70.4	67.9	69.1

In the meanwhile, we also adjust the boundaries of the expanded form according to the abbreviation, if necessary. Finally, the expanded form and its abbreviation are stored in the recognized list of biomedical entity names from the document to help the resolution of forthcoming occurrences of the same abbreviation in the document.

4. Experimental results

We evaluate our PowerBioNE system on GENIA V1. 1 and V3.0 using precision/recall/*F*-measure. For each evaluation, we select 20% of the corpus as the held-out test data and the remaining 80% as the training data. All the experiments are cross-validated five times and the evaluations are averaged over the held-out test data. For nested entity name resolution, an average of 59 and 97 rules are extracted from the nested entity names in the training data of GENIA V1.1 and V3.0, respectively. For POS, all the POS taggers are trained on the training data with POS imported from the corresponding GENIA V3.02p with POS annotated.

Table 4 shows the performance of our system on GENIA V1.1 and V3.0, and the comparison with that of the best reported system [24]. It shows that our system achieves an *F*-measure of 69.1 on GENIA V1.1 and an *F*-measure of 71.2 on GENIA V3.0, respectively, without help of any dictionaries. It also shows that our system outperforms [24] by 6.9 in *F*-measure on GENIA V1.1 and 4.6 in *F*-measure on GENIA V3.0. This is due to the superiority of the SVM plus sigmoid in our system over the back-off approach in [24] and the novel name alias feature. Table 5 shows the contributions of various features (including the word itself) and post processing modules in our system on the GENIA V3.0 corpus by leaving one feature or one post processing module at a time. It shows that

- The novel name alias feature contributes 1.2 in *F*-measure. This means that the SVM plus sigmoid approach in our system outperforms the back-off approach in [24] by contributing 3.4 (4.6−1.2) more in *F*-measure.

Table 5 Contributions of various features (including the word itself) and post processing modules in our system on GENIA V3.0: decrease in *F*-measure by leaving one feature or one post processing module at a time (five-fold cross-validation)

Features/post processing modules	*F*
Word itself	−12.6
Word formation pattern feature	−16.3
Morphological pattern feature	−1.3
Part-of-speech feature	−5.8
Head noun trigger feature	−2.1
Name alias feature	−1.2
Nested entity name resolution	−3.4
Abbreviation resolution	−2.1

- Other features contribute quite differently: (1) the word formation pattern feature contributes most by 16.3 in *F*-measure; (2) the word itself contributes second by 12.6 in *F*-measure; (3) the POS feature contributes third by 5.8 in *F*-measure; (4) the morphological pattern feature slightly improve the performance by 1.3 in *F*-measure; (5) the head noun trigger feature slightly improve the performance by 2.1 in *F*-measure.
- The nested entity name resolution contributes 3.4 in *F*-measure.
- The abbreviation resolution contributes 2.1 in *F*-measure.

One important question is about the performance of different entity classes. Table 6 shows the performance of the major biomedical entity classes on GENIA V3.0. Of particular interest, our system achieves an *F*-measure of 77.8 on the class ''*Protein*''. It shows that the performance varies a lot among different entity classes. One reason may be due to different difficulties in recognizing different entity classes. Another reason may be due to the different numbers of instances in different entity classes. Though GENIA V3.0 provides a good

Table 6 Performance of different entity classes on GENIA V3.0 (five-fold cross-validation)

Entity class	Number of instances in the training data	*F*
Cell type	6034	81.8
Lipid	1602	68.6
Multi-cell	1463	78.1
Protein	21380	77.8
DNA	7538	70.8
Cell line	3216	68.5
RNA	695	56.2
Virus	873	67.2

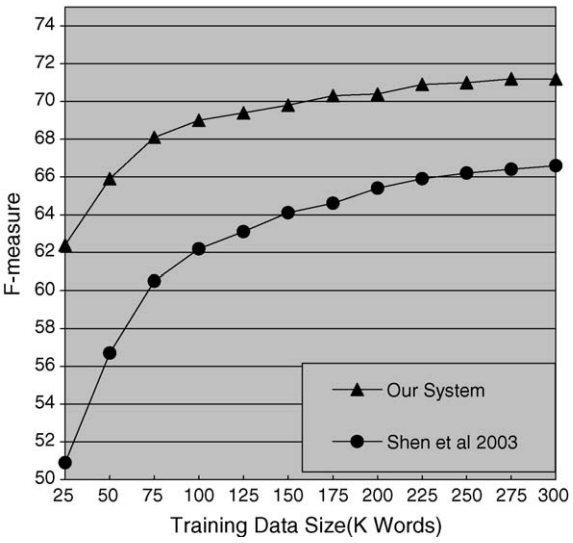

Fig. 1 Impact of different training data sizes on GENIA V3.0.

basis for named entity recognition in the biomedical domain and probably the best available, it has clear bias. Table 5 shows that, while GENIA V3.0 is of enough size for recognizing some of the major classes, such as *''Protein''*, *''Cell type''*, *''Cell line''*, *''Lipid''*, etc., it is of limited size in recognizing other classes, such as *''RNA''*.

Another important question is how the performance varies as the training data size changes? This issue relates with the portability of the system and its possible performance improvement if a much larger training corpus is available. The experimental result is shown in Fig. 1 using GENIA V3.0. Here, all the systems including respective features are modified accordingly based on the used part of the training corpus. Fig. 1 shows that further increasing the size of the training data beyond 300K words tends to only slightly improves the performance in our system while [24] is much more data driven. It also shows that our system has a turning point at the training data size of about 75K with an *F*-measure of about 68 while [24] has a turning point at about 250K with an *F*-measure of about 66. This means that our system can reach much higher stable performance much faster than [24]. This suggests that our system provides a potential for much better portability than the best reported system in [24].

5. Error analysis

In order to further evaluate our system and explore possible improvement, we have implemented an error analysis. This is done by randomly choosing 100 errors from our recognition results. During the error analysis, we find many errors are due to the strict annotation scheme and the annotation inconsistence in the GENIA corpus, and can be considered acceptable. Therefore, we will also examine the *acceptable F*-measure of our system, in particular, the *acceptable F*-measure on the *''protein''* class.

All the 100 errors are classified as follows:

- *Left boundary errors* (14): They include the errors with correct class identification, correct right boundary detection and only wrong left boundary detection. We find that most of such errors come from the long and descriptive naming convention. We also find that 11 of 14 errors are acceptable and ignorance of the descriptive words often does not make a much difference for the entity names. In fact, it is even hard for biologists to decide whether the descriptive words should be a part of the entity names, such as *''normal''*, *''activated''*, etc. In particular, 4 of 14 errors belong to the *''protein''* class. Among them, two errors are acceptable, e.g. *''classical <PROTEIN>1,25 (OH) 2D3 receptor </PROTEIN>''* $\Rightarrow$ *''<PROTEIN> classical 1,25 (OH) 2D3 receptor </PROTEIN>''* (with format of ''annotation in the corpus $\Rightarrow$ identification made by our system''), while the other two are unacceptable, e.g. *''<PROTEIN>viral transcription factor </PROTEIN>* $\Rightarrow$ *viral <PROTEIN> transcription factor </PROTEIN>''*.
- *Nested entity name errors* (16): They include the errors when processing nested entity names. We find that most of such errors come from annotation inconsistence in the GENIA corpus: In some cases, only embedded entity names are annotated while in other cases, embedded entity names are not annotated. Our system tends to annotate both embedded entity names and whole entity names. Among them, we find that 13 of 16 errors are acceptable. In particular, two of 16 errors belong to the *''protein''* class and both are acceptable, e.g. *''<DNA>NF kappa B binding site</DNA>''* $\Rightarrow$ *''<DNA><PROTEIN>NF kappa B</PROTEIN> binding site</DNA>''*.
- *Misclassification errors* (18): They include the errors with wrong class identification, correct right boundary detection and correct left boundary detection. We find that this kind of errors mainly comes from the sense ambiguity of biomedical entity names and is very difficult to disambiguate. Among them, 8 errors are related with the *''DNA''* class and six errors are related with the *''Cell Line''* and *''Cell type''* classes. We also find that only three of 18 errors are acceptable. In particular, there are six errors related to the *''protein''* class. Finally, we find

that all the six errors are caused by misclassification of the ''*DNA*'' class to the ''*protein*'' class and all of them are unacceptable, e.g. ''*<DNA>type I IFN<DNA>*'' ⇒ ''*<PROTEIN>type I IFN</PROTEIN>*''.

- *True negative* (23): They include the errors by missing the identification of biomedical entity names. We find that 13 errors come from the ''*other*'' class and 10 errors from the ''*protein*'' class. We also find that the GENIA corpus annotates some general noun phrases as biomedical entity names, e.g. ''protein'' in ''the protein'' and ''cofactor'' in ''a cofactor''. Finally, we find that 11 of 23 errors are acceptable. In particular, nine of 23 errors related to the ''*protein*'' class. Among them, three errors are acceptable, e.g. ''*the <PROTEIN>protein</PROTEIN>*'' ⇒ ''*the protein*'', while the other six are unacceptable, e.g. ''*<PROTEIN>80 kDa</PROTEIN>*'' ⇒ ''*80 kDa*''.
- *False positive* (15): They include the errors by wrongly identifying biomedical entity names, which are not annotated in the GENIA corpus. We find that nine of 15 errors come from the ''*other*'' class. This suggests that the annotation of the ''*other*'' class is much lacking in consistency and the most problematic in the GENIA corpus. We also find that seven of 15 errors are acceptable. In particular, two of 15 errors are related to the ''*protein*'' class and both are acceptable, e.g. ''*affinity sites*'' ⇒ ''*<PROTEIN>affinity sites</PROTEIN>*''.
- *Miscellaneous* (14): They include all the other errors, e.g. combination of the above errors and the errors caused by parentheses. We find that only one of 14 errors is acceptable. We also find that, among them, two errors are related to the ''*protein*'' class and both are unacceptable, e.g. ''*<PROTEIN>17 amino acid epitope</PROTEIN>*'' ⇒ ''*17 <RNA>amino acid epitope</RNA>*''.

From the above error analysis, we find that about half (46/100) of errors are acceptable and can be avoided by a more flexible annotation scheme (e.g. regarding the modifiers in the left boundaries) and consistent annotation (e.g. in the annotation of the ''*other*'' class and nested entity names). In particular, about one third (9/25) of errors are acceptable on the ''*protein*'' class. This means that the acceptable *F*-measure can reach about 84.4 on the 23 classes of GENIA V3.0. In particular, the acceptable *F*-measure on the ''*protein*'' class is about 85.8. In addition, this performance is achieved without using any extra resources (e.g. dictionaries). With help of extra resources, we think an acceptable *F*-measure of near 90 can be achieved in the near future.

6. Related work

Previous approaches in biomedical name recognition typically use some domain specific heuristic rules and heavily rely on existing dictionaries in a very limited biomedical domain [17,16,21]. Fukuda et al. [17] uses some heuristic rules to identify ''*protein*'' names in SH3 protein domain and evaluation on 30 annotated MEDLINE abstracts shows precision of 95.22% and recall of 97.40%. Proux et al. [16] uses finite state technology to detect ''*gene*'' names by using lexical and morphological knowledge. It achieves precision of 91.4% and recall of 94.4% on a small corpus of 1200 sentences from Flybase [40], and precision of about 70% on a larger corpus of 25,000 abstracts from MEDLINE. Gaizauskas et al. [21] derives their system from an existing system in the newswire domain (MUC) and applies it in two projects: extraction of enzymes and metabolic pathways (EMPathIE) and extraction of protein structure (PASTA). Their system mainly uses lexicons, morphological cues, such as suffixes, and hand-coded rules. Evaluation of biomedical named entity recognition on 6 full journal articles in EMPathIE task achieves precision of 86% and recall of 68% on 10 named entity classes, such as ''*compound*'', ''*element*'', ''*enzyme*'', etc. while evaluation on 52 MEDLINE abstracts in PASTA task achieves precision of 94% and recall of 88% on 13 named entity classes, such as ''*protein*'', ''*species*'', ''*residue*'', etc. Although these rule-based systems seem quite promising, they lack the ability of adaptation to new named entity classes in the biomedical domain. Once a new class is required to identify, a set of rules for this class has to be generated manually. The more classes, the more ambiguous and difficult to construct consistent rules. Moreover, such approaches only work well in very limited biomedical domains with biomedical name dictionaries of good coverage. Finally, these systems heavily depend on well-developed dictionaries.

The current trend is to apply machine-learning approaches in biomedical name recognition, largely due to the development of the GENIA corpus. Typical explorations include [18,20,14,22,23,41,24,25]. Nobata et al. [18], Collier et al. [20] and Takeuchi and Collier [14] use a preliminary version of the GENIA corpus which contains 100 abstracts and identifies 10 named entity classes, such as ''protein'', ''DNA'', ''RNA'', ''cell line'', ''cell type'', etc. Nobata et al. [18] uses the decision tree

and incorporates POS, character information, and domain specific word lists. The experiment shows that it achieves an F-measure of about 56. Collier et al. [20] applies a linear interpolated HMM and incorporates surface word itself and character information. It achieves an F-measure of 72.8. Takeuchi and Collier [14] applies SVM and incorporates surface word itself, character information and pervious word class tags (−3 to +3). It achieves an *F*-measure of 71.8. Kazama et al. [22] also applies SVM and incorporates a rich feature set, including word, POS, prefix, suffix, previous class, word cache and HMM state. Experimentation on GENIA V1.1 of 23 classes shows an *F*-measure of 54.4, compared with our 62.2 on the same GENIA V1.1. Tsuruoka and Tsujii [41] applies a dictionary-based approach and a naïve Bayes classifier to filter out false positives. It only evaluates against the ''*protein*'' class in GENIA V3.0, and receives an *F*-measure of 70.2 with help of a large dictionary, compared with our 75.8 on the same class without help of any dictionaries. Lee et al. [23] uses a two-phase SVM-based recognition approach and incorporates word formation pattern and part-of-speech (existing POS in the corpus). The evaluation on GENIA V3.0 shows an *F*-measure of 66.5 over 22 classes except the ''*other*'' class with help of an entity name dictionary. Shen et al. [24] and Zhou et al. [25] explore a rich feature set such as word formation pattern, morphological pattern, POS, head noun trigger, special verb trigger and name alias feature. All the features are integrated via a HMM-based approach. Finally, two post-processing modules (nested entity name resolution and abbreviation resolution) are presented to further improve the performance. Evaluation shows *F*-measures of 62.2 and 66.6 on GENIA V1.1 and V3.0, respectively.

7. Conclusion

In the paper, we describe our MIIM (mutual information independence model)-based named entity recognition system in the biomedical domain, named PowerBioNE. Various lexical, morphological, syntactic, semantic and discourse features are incorporated to cope with the special phenomena in biomedical name recognition. In addition, a SVM plus sigmoid is proposed to effectively resolve the data sparseness problem in the MIIM. Finally, we present two post-processing modules to deal with nested entity name and abbreviation phenomena to further improve the performance. In this way, a biomedical name recognition system with better performance and better portability is achieved.

The main contributions of our work are the novel name alias feature in the biomedical domain, the SVM plus sigmoid approach in the effective resolution of the data sparseness problem in our system and its integration with the mutual information independence model (MIIM).

In the near future, we will further improve the performance by further investigating the conjunctive and disjunctive structures, the synonym phenomenon, and the use of extra resources (e.g. dictionary).

References

[1] MUC6, Proceedings of the Sixth Message Understanding Conference (MUC-6), Morgan Kaufmann Publishers, Inc., Columbia, Maryland, 1995.
[2] MUC7, Proceedings of the Seventh Message Understanding Conference (MUC-7), Morgan Kaufmann Publishers, Inc., Fairfax, Virginia, 1998.
[3] CoNLL, Proceedings of the Sixth Conference on Computational Natural Language Learning, Morgan Kaufmann Publishers, Inc., Taipei, Taiwan, 2002.
[4] CoNLL, Proceedings of the Seventh Conference on Computational Natural Language Learning, Morgan Kaufmann Publishers, Inc., Edmonton, Canada, 2003.
[5] S. Miller, M. Crystal, H. Fox, L. Ramshaw, R. Schwartz, R. Stone, R. Weischedel, The Annotation Group, BBN: description of the SIFT system as used for MUC-7, in: Proceedings of the Seventh Message Understanding Conference (MUC-7), Fairfax, Virginia, 1998.
[6] M. Bikel Daniel, R. Schwartz, M. Weischedel Ralph, An algorithm that learns what's in a name, Mach. Learn 34 (3) (1999) 211—231, Special Issue on NLP.
[7] G.D. Zhou, J. Su, Named entity recognition using an HMM-based chunk tagger, in: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), 2002, pp. 473—480.
[8] S. Miller, Guinness, A. Zamanian, Name Tagging with Word Clusters and Discriminative Training, HLT'2004, Boston, MA, 2004, pp. 347—342.
[9] A. Borthwick, J. Sterling, E. Agichtein, R. Grishman, NYU: description of the MEME named entity system as used in MUC-7, in: Proceedings of the Seventh Message Understanding Conference (MUC-7), Fairfax, Virginia, 1998.
[10] A. Borthwick, A maximum entropy approach to named entity recognition, Ph.D. thesis, New York University, 1999.
[11] H.L. Chieu, H.T. Ng, Named entity recognition: a maximum entropy approach using global information, in: Proceedings of the 19th International Conference on Computational Linguistics (COLING'2002), Taipei, Taiwan, 2002, pp. 190—196.
[12] H.L. Chieu, H.T. Ng, Named entity recognition with a maximum entropy approach, in: Proceedings of CoNLL-2003, Edmonton, Canada, 2003, pp. 160—163.
[13] A. McCallum, W. Li, Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons, in: Proceedings of CoNLL-2003, Edmonton, Canada, 2003, pp. 188—191.
[14] K. Takeuchi, N. Collier, Use of support vector machines in extended named entity recognition, in: Proceedings of the Sixth Conference on Natural Language Learning (CoNLL 2002), 2002, pp. 119—125.

[15] S.W. Bennett, C. Aone, C. Lovell, Learning to tag multilingual texts through observation, in: Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP'1996), Providence, Rhode Island, 1996, pp. 109—116.
[16] D. Proux, F. Rechenmann, L. Julliard, V. Pillet, B. Jacq, Detecting gene symbols and names in biological texts: a first step toward pertinent information extraction, in: Proceedings of the Genome Information Series Workshop Genome Information, 1998, pp. 72—80.
[17] K. Fukuda, T. Tsunoda, A. Tamura, T. Takagi, Toward information extraction: identifying protein names from biological papers, in: Proceedings of the Pacific Symposium on Biocomputing'98 (PSB'98), 1998, pp. 707—718.
[18] C. Nobata, N. Collier, J. Tsujii, Automatic term identification and classification in biology texts, in: Proceedings of the Fifth NLPRS, 1999, pp. 369—374.
[19] C. Nobata, N. Collier, J. Tsujii, Comparison between tagged corpora for the named entity task, in: Proceedings of the Workshop on Comparing Corpora (at ACL'2000), 2000, pp. 20—27.
[20] N. Collier, C. Nobata, J. Tsujii, Extracting the names of genes and gene products with a hidden Markov model, in: Proceedings of COLING, 2000, pp. 201—207.
[21] R. Gaizauskas, G. Demetriou, K. Humphreys, Term recognition and classification in biological science journal articles, in: Proceedings of the Computational Terminology for Medical and Biological Applications Workshop of the Second International Conference on NLP, 2000, pp. 37—44.
[22] J. Kazama, T. Makino, Y. Ohta, J. Tsujii, Tuning support vector machines for biomedical named entity recognition, in: Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain (at ACL'2002), 2002, pp. 1—8.
[23] K.J. Lee, Y.S. Hwang, H.C Rim, Two-phase biomedical NE Recognition based on SVMs, in: Proceedings of the ACL'2003 Workshop on Natural Language Processing in Biomedicine, Sapporo, Japan, 2003, pp. 33—40.
[24] D. Shen, J. Zhang, G.D. Zhou, J. Su, C.L. Tan, Effective adaptation of a hidden markov model-based named entity recognizer for biomedical domain, in: Proceedings of the ACL'2003 Workshop on Natural Language Processing in Biomedicine, Sapporo, Japan, 11 July, 2003, pp. 49—56.
[25] G.D. Zhou, J. Zhang, J. Su, D. Shem, C.L. Tan, Recognizing names in biomedical texts: a machine learning approach, Bioinformatics 20 (7) (2004) 1178—1190.
[26] R. Weischedel, M. Meteer, R. Schwartz, L. Ramshaw, J. Palmucci, Coping with ambiguity and unknown words through probabilistic methods, Computational Linguistics 19 (2) (1993) 359—382.
[27] B. Merialdo, Tagging English text with a probabilistic model, Computational Linguistics 20 (2) (1994) 155—171.
[28] K.W. Church, A stochastic pars program and noun phrase parser for unrestricted text, in: Proceedings of the Second Conference on Applied Natural Language Processing (ANLP'1998), Austin, Texas, 1998.
[29] T. Brants, W. Skut, B. Krenn, Tagging grammatical functions, in: EMNLP'1997, Brown University, 1997.
[30] W. Skut, T. Brants, Chunk tagger-statistical recognition of noun phrases, in: Proceedings of the ESSLLI'98 workshop on Automatic Acquisition of Syntax and Parsing, University of Saarbrucken, Germany, 1998.
[31] A.J. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Trans. Inform. Theory (1967) 260—269.
[32] L. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, IEEE 72 (2) (1989) 257—285.
[33] T. Ohta, Y. Tateisi, J. Kim, H. Mima, J. Tsujii, The GENIA corpus: an annotated research abstract corpus in molecular biology domain, in: Proceedings of HLT2002, 2002.
[34] A.S. Schwartz, M.A. Hearst, A simple algorithm for identifying abbreviation definitions in biomedical text, in: Proceedings of the Pacific Symposium on Biocomputing (PSB 2003) Kauai, 2003.
[35] C. Jacquemin, Spotting and Discovering Terms through Natural Language Processing, MIT Press, Cambridge, 2001.
[36] Chen, Goodman, An Empirical Study of Smoothing Techniques for Language Modeling, in: Proceedings of the 34th Annual Meeting of the Association of Computational Linguistics (ACL'1996), Santa Cruz, California, USA, 1996, pp. 310—318.
[37] T. Joachims, Text categorization with support vector machines: learning with many relevant features, in: Proceedings of the European Conference on Machine Learning (ECML'1998), Chemnitz, Germany, 21—23 April, 1998.
[38] V. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, NY, USA, 1995.
[39] J. Platt, Probabilistic Outputs for Support Vector Machines and comparisions to regularized Likelihood Methods, MIT Press, 1999.
[40] The Flybase Consortium, Flybase-A *Drosophila* database, Nucleic Acids Res. 26 (1998) 85—88.
[41] Y. Tsuruoka, J. Tsujii, Boosting precision and recall of dictionary-based protein name recognition, in: Proceedings of the ACL'2003 Workshop on Natural Language Processing in Biomedicine, Sapporo, Japan, 2003, pp. 41—48.