Artificial Intelligence in Medicine (2005) **33**, 125—137

ARTIFICIAL INTELLIGENCE IN MEDICINE

http://www.intl.elsevierhealth.com/journals/aiim

Bio-medical entity extraction using support vector machines

Koichi Takeuchi[a,*], **Nigel Collier**[b,1]

[a]*Okayama University, 3-1-1 Tsushima-naka, Okayama-shi, Okayama 700-8530, Japan*
[b]*National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan*

Received 13 December 2002; received in revised form 9 July 2004; accepted 15 July 2004

KEYWORDS
Text mining;
Support vector machines;
Machine learning;
Multi-classifier;
Natural language processing;
Named entity;
MEDLINE

Summary

Objective: Support vector machines (SVMs) have achieved state-of-the-art performance in several classification tasks. In this article we apply them to the identification and semantic annotation of scientific and technical terminology in the domain of molecular biology. This illustrates the extensibility of the traditional named entity task to special domains with large-scale terminologies such as those in medicine and related disciplines.
Methods and materials: The foundation for the model is a sample of text annotated by a domain expert according to an ontology of concepts, properties and relations. The model then learns to annotate unseen terms in new texts and contexts. The results can be used for a variety of intelligent language processing applications. We illustrate SVMs capabilities using a sample of 100 journal abstracts texts taken from the {*human, blood cell, transcription factor*} domain of MEDLINE.
Results: Approximately 3400 terms are annotated and the model performs at about 74% F-score on cross-validation tests. A detailed analysis based on empirical evidence shows the contribution of various feature sets to performance.
Conclusion: Our experiments indicate a relationship between feature window size and the amount of training data and that a combination of surface words, orthographic features and head noun features achieve the best performance among the feature sets tested.
© 2004 Elsevier B.V. All rights reserved.

1. Introduction

With the rapid growth in the number of published papers in the scientific fields such as medicine there has been growing interest in the application of information extraction (IE) [1,2] to help solve some

* Corresponding author. Tel.: +81 86 251 8178; fax: +81 86 251 8178.
E-mail addresses: koichi@cl.it.okayama-u.ac.jp (K. Takeuchi), collier@nii.ac.jp (N. Collier)
URL: research.nii.ac.jp/~collier.
[1] Tel. +81 3 4212 2536; fax: +81 3 3556 1916.

0933-3657/$ — see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.artmed.2004.07.019

of the problems that are associated with information overload. IE can benefit the medical sciences by enabling the automatic extraction of facts related to *prototypical* events such as those contained in patient records or research articles regarding molecular processes and their effect on human health. These facts can then be used to populate databases, aid in searching or document summarization and a variety of tasks which require the computer to have an intelligent understanding of the contents inside a document.

Our aim here is to show a state-of-the-art method for identifying and classifying technical terminology. According to the theoretical and practical work done on terminology extraction [3,4] we can broadly define 'term' and 'terminology' as "a lexical unit representing one or more words that represents a concept in a domain".

The term extraction task is an extension of the *named entity* task defined by the DARPA-sponsored Message understanding conferences (MUCs) [5] and is aimed at acquiring the shallow semantic building blocks that contribute to a high level understanding of the text. Although our study here looks at shallow semantics that can be captured using IE, our basic goal is to join this with deep semantic representations so that computers can obtain an understanding of selected events in a text using logical inference and reasoning. The scenario is that human experts will create taxonomies and axioms (*ontologies*) and by providing a small set of annotated examples, machine learning can take over the role of instance capturing though information extraction technology.

Recent studies into the use of supervised learning-based models for the named entity task have shown that models based on hidden Markov models (HMMs) [6,7], decision trees [8], and maximum entropy [9] are much more generalisable and adaptable to new classes of words than systems based on hand-built patterns (including wrappers) and domain specific heuristic rules such as [10].

The method we use is based on support vector machines (SVMs) [11], a state-of-the-art model that has achieved new levels of performance in many classification tasks. In previous work, we have shown SVMs to be superior to several other commonly used machine learning methods for named entity in previous experiments such as HMMs [12] and C4.5 [13].[2] On the Bio1 training set which we use in our experiments, the best reported *F*-scores for accuracy of annotation were 74.23 (SVM), 73.1 (HMM), and 69.4 (C4.5). A comparison on the MUC-6 dry run data set given in [13] confirmed the trend for HMM and C4.5 with 74.2 (HMM) and 68.5 (C4.5). This paper explores the underlying SVM model and shows through detailed empirical analysis the key features and parameter settings.

[2] This study was conducted between one of the authors (Collier) and Nobata and Tsujii

To show the application of SVMs to term extraction in unstructured texts related to the medical sciences we are using a collection of abstracts from PubMed's MEDLINE [14]. The MEDLINE database is an online collection of abstracts for published journal articles in biology and medicine and contains more than nine million articles. The collection we use in our tests is a controlled subset of MEDLINE obtained using three search keywords in the domain of molecular biology. From the retrieved abstracts 100 were randomly chosen for annotation by a human expert according to classes in a small core domain specific ontology.

In the remainder of this paper in Section 2, we outline the background to the task and the data set we are using; in Section 3, we described the basic advantages of SVMs and the formal model we are using as well as implementation specific issues such as the choice of feature set and report experimental results. In Section 4, we provide extensive results and a discussion of four sets of experiments we conducted that show the best feature sets and parameter settings in our sample domain.

2. Material

The names that we are trying to extract fall into a number of categories that are outside the definitions used for the traditional named-entity task used in MUC. For this reason, we consider the task of term identification and classification to be an *extended named entity* task (NE+) in which the goal is to find types as well as individuals and where the term classes belong to an explicitly defined *ontology* in a domain [15]. By *domain*, we refer to a notion of consensus among a group of people who share a view on the structure of knowledge in a particular area. The terms are representative of the domain and can be considered to form the vocabulary for the domain.

The use of an ontology allows us to associate human-readable terms with a set of computer-readable classes, relations, properties and axioms [16]. The hierarchical taxonomic relations formed between classes in the ontology provide a potential for logical inference in a way that non-hierarchical disjoint relations commonly used in MUC-style named entity do not.

Considering types versus individuals, there are several issues that may mean that NE+ is more

Table 1 Markup classes used in Bio1 with the number of word tokens for each class

Class	#	Examples	Description
PROTEIN	2125	*hUBC9,ATF2,* *MAP kinase/Erk kinase, 4-1BB*	Proteins, protein groups, Families, complexes and substructures.
DNA	358	*Cyclic AMP-responsive element*	DNAs, DNA groups, regions and genes
RNA	30	*TAR, transactivation responsive*	RNAs, RNA groups, regions and genes
SOURCE.cl	93	*Leukemic T cell line Kit225*	Cell line
SOURCE.ct	417	*Peripheral blood T lymphocytes CEM cells,*	Cell type
SOURCE.mo	21	*Schizosaccharomyces pombe*	Mono-organism
SOURCE.mu	64	*Mammalian, Drosophila*	Multiorganism
SOURCE.vi	90	*Human adenoviruses,* *Human immunodeficiency virus*	Viruses
SOURCE.sl	77	*Nuclear, nuclei, membrane*	Sublocation
SOURCE.ti	37	*Central nervous system*	Tissue

challenging than NE. The most important is the number of variants of NE+ expressions due to graphical, morphological, shallow syntactic and discourse variations. For example, the use of head sharing combined with embedded abbreviations in *unliganded (apo)- and liganded (holo)-LBD*. Such expressions will require syntactic analysis beyond simple noun phrase chunking if they are to be successfully captured. NE+ expressions may also require richer contextual evidence than is needed for regular NEs—for example knowledge of the head noun or the predicate—argument structure.

The particular difficulties with identifying and classifying terms in scientific and technical domains are the size of the vocabulary [17], an open growing vocabulary [18], irregular naming conventions as well as extensive cross-over in vocabulary between classes. The irregular naming arises in part because of the number of researchers and practitioners from different fields who are working on the same knowledge discovery area as well as the large number of entities that need to be named. Despite the best efforts of major journals to standardize the terminology, there is also a significant problem with synonymy so that often an entity has more than one name that is widely used. In molecular biology, for example, class cross-over of terms may arise because many DNA and RNA are named after the protein with which they transcribe. This *semantic ambiguity* which is dependent on often complex contextual conditions is one of the main reasons why we need learnable models and why it is difficult to re-use existing term lists and vocabularies such as MeSH [19], UMLS [17] or those found in databases such as SwissProt [20]. An additional obstacle to re-use is that the classification scheme used within an existing thesaurus or database may not be the same as the one in the users' ontology which may change from time to time as the consensus view of the structure of knowledge is refined.

Our work has focussed on identifying NE+ expressions belonging to the classes shown in Table 1 which are all taken from the domain of molecular biology. Example sentences from a marked up abstract are given in Fig. 1. The ontology [21] that underlies this classification scheme describes a simple top-level model which is almost flat except for the *source* class which shows places where genetic activity occurs and has a number of sub-types. Further discussion of our use of deep semantic structures in the ontology is given elsewhere [22] and we will now focus our attention on the machine learning model used to capture low level semantics.

The training set we used in our experiments called Bio1[3] consists of 100 MEDLINE abstracts, marked up in XML by a doctoral-qualified domain expert for the name classes given in Table 1. The number of named entities that were marked up by class are also given in Table 1 and the total number of words in the corpus is 29940. The abstracts were chosen from a sub-domain of molecular biology that we formulated by searching under the terms *human*, *blood cell*, *transcription factor* in the PubMed database. An example can be seen in Fig. 1

3. Method

In this section, we give a brief summary of SVMs covering separable and non-separable cases and the basic motivating ideas. For a tutorial on SVMs we refer the reader to [23] and details of their formulation can be found in [24].

The named entity task can be formulated as a type of classification task for supervised learning in which the computer takes a set of l input-output pairs $Z = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ and attempt to con-

[3] Available from http://www.research.nii.ac.jp/~collier/resources/bio1.1.xml.

TI - Differential interactions of <NAME cl="PROTEIN">Rel</NAME>- <NAME cl="PROTEIN">NF-kappa B</NAME> complexes with <NAME cl="PROTEIN">I kappa B alpha</NAME> determine pools of constitutive and inducible <NAME cl="PROTEIN">NF-kappa B</NAME> activity.
AB - The <NAME cl="PROTEIN">Rel</NAME>- <NAME cl="PROTEIN">NF-kappa B</NAME> family of transcription factors plays a crucial role in the regulation of genes involved in inflammatory and immune responses. We demonstrate that in vivo, in contrast to the other members of the family, <NAME cl="PROTEIN">RelB</NAME>associates efficiently only with <NAME cl="PROTEIN">NF-kappa B1 </NAME> (<NAME cl="PROTEIN">p105-p50</NAME>) and <NAME cl="PROTEIN">NF-kappa B2</NAME> (<NAME cl="PROTEIN">p100-p52</NAME>), but not with <NAME cl="PROTEIN">cRel</NAME> or <NAME cl="PROTEIN">p65</NAME>. The <NAME cl="PROTEIN">RelB </NAME>- <NAME cl="PROTEIN">p52 </NAME>heterodimers display a much lower affinity for <NAME cl="PROTEIN">I kappa B alpha</NAME> than <NAME cl="PROTEIN">RelB</NAME>- <NAME cl="PROTEIN">p50</NAME> heterodimers or <NAME cl="PROTEIN">p65</NAME> complexes. However, similarly to the other <NAME cl="PROTEIN">Rel</NAME>- <NAME cl="PROTEIN">NF-kappa B</NAME> complexes, <NAME cl="PROTEIN">RelB</NAME>- <NAME cl="PROTEIN">p52</NAME> can upregulate the synthesis of <NAME cl="PROTEIN">I kappa B alpha</NAME> leading to the <NAME cl="SOURCE.sl">cytoplasmic</NAME> trapping of dimers which have a higher affinity for the inhibitor. We suggest that a hierarchy of interactions between <NAME cl="PROTEIN">I kappa B alpha</NAME> and the different <NAME cl="PROTEIN">Rel</NAME>- <NAME cl="PROTEIN">NF-kappa B</NAME>complexes governs their cellular distribution. This results in the presence of two distinct pools of <NAME cl="PROTEIN">NF-kappa B </NAME> activity which differ in their composition: one a constitutive nuclear and the other an inducible cytoplasmic activity.

Figure 1 Example MEDLINE sentence marked up in XML for molecular biology named-entities.

struct a classification function $f : X \rightarrow \mathrm{R}$ that maps N dimensional input pattern vectors $x \in X$ onto a set of labels $y \in Y$. In the case of a two class pattern classifier, the labels are simply $Y = \{-1, +1\}$, i.e.

$$(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \subset \mathrm{R}^N \times \{\pm 1\} \tag{1}$$

The goal is then to learn $f \in F$ which minimizes the error $(f(x) \neq y)$ on randomly chosen unseen examples. These unseen examples are drawn from the total example data. From now we will consider the example data to be partitioned into a set of examples unseen by the model during training known as *testing data* and the remainder which is used for learning known as training data. We consider the examples to be generated by an unknown probability distribution $P(x, y)$ and then make the fundamental assumption that the testing data and training data come from the same underlying distribution. The average error committed by function f_α on testing data is called the *risk*.

$$R(\alpha) = \int \frac{1}{2} |f_\alpha(x) - y| \mathrm{d}P(x, y) \tag{2}$$

where $\alpha \in \Lambda$ are Lagrange parameters to be estimated in learning.

Since both P and hence R are unknown we attempt to minimize the *empirical risk* calculated using testing data

$$R_{\mathrm{empircal}}(\alpha) = \frac{1}{l} \sum_{i=1}^{l} \frac{1}{2} |f_\alpha(x_i) - y_i| \tag{3}$$

Empirical risk could be accurately estimated and minimized if we had enough example data. The variance of this estimate reduces as the size of the training data increases.

3.1. Cross validation

Since the example data is almost always limited in quantity it is common practice to partition the example data set multiple times, called *folds*, so that each example is used in different folds as training data and as testing data. This approach which we use in our experiments is known as *cross validation* and generally allows us to obtain more reliable estimates of model error than would be possible by just partitioning the example set once. In the extreme case where the number of folds equals the number of examples (l) this approach defaults to the leave-one-out approach which uses $l-1$ examples for training and 1 example for testing.

3.2. Support vector machines

There are several effective and well-studied learning algorithms available for the classification task such as naive Bayes, logistic regression, transformation-based error-driven learning and decision trees. SVMs are a recent model [11,25] that has been the focus of intensive research in machine learning due to its capacity to learn effectively from large *sparse* feature sets. SVMs realize the idea of using a kernel function Φ to embed the input space $x \in \mathrm{R}^N$ in a high dimensional hypothesis space H where it is possible to form a linear discrimination function that maximizes the geometric margin between the positive and negative classes. This is often more effective than other methods which attempt to construct a possibly far more complex function in the input space itself. The basic approach is shown in Fig. 2.

Returning to the discussion of risk, as a consequence of the limited number of training examples we can expect, statistical methods have been developed which attempt to minimize *true risk*, as opposed to the empirical risk of Eq. (3). Since the real value of true risk cannot be found we have to estimate bounds for it from training data. SVMs use one such statistical method called the Structural Risk Minimization principal [26] which is based on the fact that for any $\alpha \in \Lambda$ and $l > h$, with a probability of at least $1-\eta$ the following equation forms the bound [27],

$$R(\alpha) \leq R_{\text{empirical}}(\alpha) + \phi\left(\frac{h}{l}, \frac{\log(\eta)}{l}\right) \tag{4}$$

where the *confidence term* Φ is defined as

$$\phi\left(\frac{h}{l}, \frac{\log(\eta)}{l}\right) = \sqrt{\frac{h(\log 2l/h + 1) - \log(\eta/4)}{l}} \tag{5}$$

This basically means that the true risk is dependent on the empirical risk and on the complexity of the learning function used as given in Eq. (5). A simple classification function will generally achieve high empirical risk and a low complexity whereas a complex function may achieve low empirical risk at the cost of high complexity, i.e. overfitting. In order to achieve balance between these two competing objectives SVMs select a function for which Eq. (5) is minimized.

The parameter h is known as the VC (Vapnik—Chervonenkis) dimension of a set of functions and is defined as the largest number of examples from a subset of X that the function H can classify correctly. The aim in SVMs is to find a hypothesis for which we can guarantee the lowest true risk. In other words, this is the probability that the hypothesis will make an error on an unseen randomly selected test example. For a given training set Z the support vector machines use the Structured Risk Minimization principal to select the function $f_{\alpha_l^\eta}$ in the subset $\{f_\alpha :$

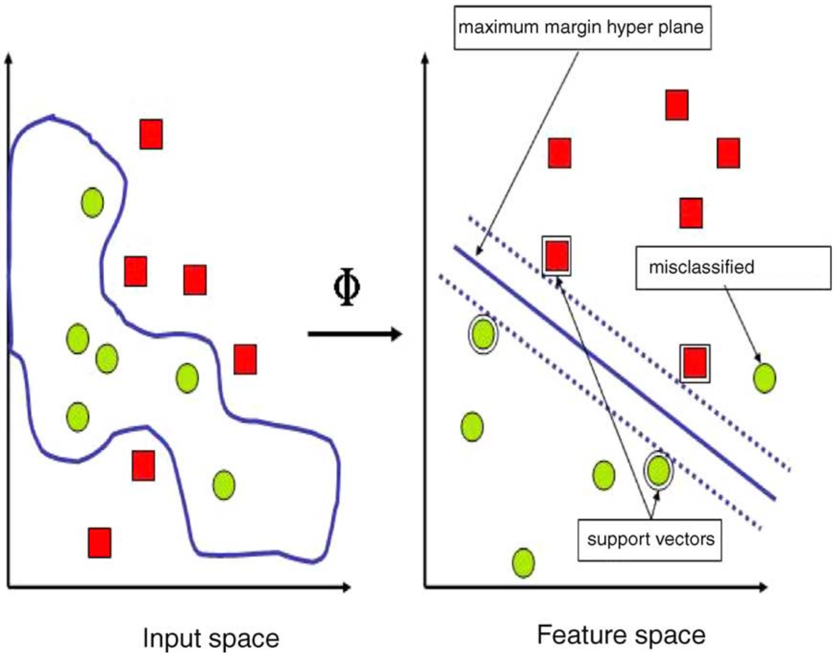

Figure 2 Mapping from input space to feature space using the kernel function Φ. The maximum margin hyperplane separating the two classes (squares and circles) is shown together with support vectors and a misclassified instance. A complex non-linear decision function in input space is also shown.

$\alpha \in \Lambda_n\}$ for which the true risk bound (given as the right hand side of Eq. (4) is approximately minimal. This is done by efficiently controlling h.

3.3. SVM algorithm

In their basic form SVMs use linear threshold functions to discriminate between two classes. This raises the question of how can they be used to capture non-linear classification functions: the answer to this is by the use of the non-linear mapping function Φ which maps the input space R^N into a hypothesis space H. Commonly used kernel functions include polynomial, sigmoid and radial basis functions.

Using the notation of [28] we now introduce some standard notation for describing SVMs. The kernel function $K(x_i, x_j)$ defines an inner product in the H space. The hyperplane decision function given by the SVM is

$$f(x) = \mathrm{sign}\left(\sum_{i=1}^{l} \alpha_i y_i \cdot K(x_i, x) + b\right) \quad (6)$$

where $b \in \mathrm{R}$ is the bias. A point x is classified as positive (or negative) if $f(x) > 0$ (or $f(x) < 0$). The Lagrange parameters α_i are estimated using quadratic optimization to maximize the following target function which is solvable in polynomial time

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad (7)$$

under the constraints that

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \quad (8)$$

and

$$0 \leq \alpha_i, i = 1, \ldots, l \quad (9)$$

The coefficients α_i defined a maximum margin hyperplane in H space where the data is mapped through a non-linear function Φ such that $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$. The use of a kernel means that it is possible to construct a linear discrimination function in H space without the prohibitive expense of explicitly mapping into it by simply finding the dual of the function and replacing the dot product with a kernel. Clearly, the complexity of data being classified determines which particular kernel should be used and more complex kernels require longer training times. The kernel function we explored in our experiments was the polynomial function $K(x_i, x_j) = \langle x_i, x_j \rangle^d$ for $d = 2$ which we found to be the best in previously reported experiments [12].

It can be seen from Eq. (7) that the number of parameters to be estimated in α never exceeds the number of examples. The influence of α_i basically means that training examples with $\alpha_i > 0$, called *support vectors*, define the maximum margin hyperplane. The optimization theory used to derive the SVM ensures that the solution to Eq. (7) is sparse, with many $\alpha_i = 0$, making the final model very compact and testing (but not training) very fast.

The formulation of a linear discrimination function with no errors, called a *hard margin SVM*, satisfies the inequality $y_i f(x_i) \geq 1$ for all examples in the training set. When the training data is separable in H space, the optimal linear threshold function will be that which maximizes the distance between the hyperplane and the closest image $\Phi(x_i)$ of vector x_i in the training data. By maximizing the margin, we ensure that the classifier minimizes redundancy and maximizes generality or in other words it reduces the risk of overfitting.

For non-separable training data one version of the SVM algorithm can misclassify some training examples (known as *outliers*). This is done using a parameter C to control the tradeoff between margin maximization and error minimization. When the training data is relatively small control of C becomes relatively more important to ensure optimal results. This approach is known as the *soft margin* algorithm [29] method and is generally preferred to using a more complex Kernel function. Using the control parameter C the constraint in Eq. (9) then becomes

$$0 \leq \alpha_i \leq C, i = 1, \ldots, l \quad (10)$$

It is shown in [29] that soft margin SVMs can be considered to be special cases of the hard margin SVMs by modifying the Kernel (Gram) matrix K as follows

$$K \leftarrow K + \frac{1}{C} I \quad (11)$$

where I is the identity matrix. Essentially, this implements a control on the influence of outliers and prevents them from having too large α values. For further details on the formulation see [24].

3.4. Implementation

We implemented our method using the TinySVM package from NAIST[4] which is an implementation of Vladimir Vapnik's SVM combined with several optimization algorithms based on SVM^{light}[30]. TinySVM can efficiently support tens of thousands of training examples and feature vectors with dimensions in the hundreds of thousands.

[4] Tiny SVM is available from http://www.cl.aist-nara.ac.jp/~taku-ku/software/TinySVM/.

The main features of the optimization algorithms are summarized below. The interested reader is referred to [30] for further details.

- The solution to the quadratic optimization problem of Eq. (7) is found by decomposing it into an active and inactive part. The active part is known as the 'working set' and is chosen using the steepest feasible descent algorithm. This ensures that the size of the SVM problem will fit into the computer's memory.
- Successive 'shrinking' is applied to the optimization problem of Eq. (7).
- An LRU cache algorithm is used to store the Kernel matrix *K*.

TinySVM extends SVM^{light} by incorporating a number of memory management features to make it more robust for large data sets. It also includes:

- Optimization for handling binary features—estimated to be twice as fast as SVM^{light}.
- A number of well-known model selection criteria including leave-one-out bound, VC dimension and Xi-Alpha estimation.

The multi-class model is built up from combining binary classifiers and then applying majority voting. This is the *p* airwise method given by Kreßel [31]. Basically for M target classes $M(M-1)/2$ binary S-VMs are constructed in which each classifier decides whether the example should belong to class *i* or *j* ($i \neq j$). Each classifier has one vote and majority voting is applied so that the class with the most votes is selected.

3.5. Generalising with features

In order for the model to be successful, it must recognize regularities in the training data that relate pre-classified examples of terms with unseen terms that will be encountered in testing. There are several important aspects to this including the power of the model, the similarity between the training and testing sets as well as the type of features used. In this section, we discuss the feature set.

Following on from previous studies in named entity, we chose a set of linguistically motivated word-level features that include surface word forms, part of speech tags using the Brill tagger [32] and orthographic features. Additionally, we used head-noun features that were obtained from pre-analysis of the training data set using the FDG shallow parser from Conexor [33]. A significant proportion of the terms in our corpus undergo a local syntactic transformations such as coordination which introduces ambiguity that needs to be resolved by shallow parsing. For example *the c- and v-rel (proto) oncogenes* and *NF-kappaB and I kappa B protein families*. In these cases, the head noun features *oncogene* and *family* would be added to each word in the constituent phrase. Head information is also needed when deciding the semantic category of a long term such as *tumor necrosis factor-alpha* which should be a PROTEIN, whereas *tumor necrosis factor (TNF)gene* and *tumor necrosis factor promoterregion* should both be types of DNA.

Table 2 shows the orthographic features that we used. Our intuition is that such features provide evidence that helps to distinguish name classes of words. Moreover, we hypothesize that such features will help the model to find similarities between known words that were found in the training set and unknown words (of zero frequency in the training set) and so overcome the unknown word problem. To give a simple example: if we know that *LMP*- 1 is a member of PROTEIN and we encounter *AP*- 1 for the first time in testing, we can make a fairly good guess about the category of the unknown word 'LMP' based on its sharing the same feature *TwoCaps* with the known word 'AP' and 'AP's known

Table 2 Orthographic features with examples

Orthographic feature	Example	Orthographic feature	Example
DigitNumber	15	CloseSquare	]
SingleCap	M	Colon	:
GreekLetter	alpha	SemiColon	;
CapsAndDigits	I2	Percent	%
TwoCaps	RalGDS	OpenParen	(
LettersAndDigits	p52	CloseParen	)
InitCap	Interleukin	Comma	,
LowCaps	kappaB	FullStop	.
Lowercase	kinases	Determiner	the
Hyphon	-	Conjunction	and
Backslash	/	Other	* + #
OpenSquare	[		

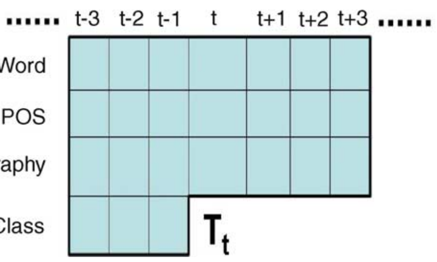

Figure 3 Lexical features and context considered by the SVM model when deciding the class tag *T* at the focus position *t*.

relationship with '-1'. The features were chosen to be as domain independent as possible, with the exception of *Hyphon* and *GreekLetter* which have particular significance for the terminology in this domain.

In the experiments, we report below we use feature vectors consisting of differing amounts of 'context' by varying the window around the focus word which is to be classified into one of the semantic classes. The full window of context considered in these experiments is ± 3 about the focus word as shown in Fig. 3. In order to show the effects of limiting context during training we tested using a number of context windows that include -10, $-1+1$, $-2+2$, $-3+2$, $-3+3$.

Word ordering is captured within the SVM software by assigning disjoint sets in the training vector to features depending on their position in the context window. Given this formation of the training vectors, experience with SVMs in other classification tasks has shown that they may be well suited to our NE+ task, i.e. using high dimensional sparse feature vectors with few irrelevant features.

4. Results and discussion

Results are given as *F*-scores [34], a common measurement for accuracy in the MUC conferences that combines recall and precision. *F*-scores are calculated using the CoNLL evaluation script[5] and are defined as $F = (2PR)/(P + R)$. where *P* denotes precision and *R* recall. *P* is the ratio of the number of correctly found NE chunks to the number of found NE chunks, and *R* the ratio of the number of correctly found NE chunks to the number of true NE chunks. All results are calculated using 10-fold cross validation.

[5] Available from http://lcg-www.uia.ac.be/conll2002/ner/.

4.1. Experiment 1: effect of training set size

The effect of total training set size is shown along the top row of Tables 3 and 4. It can be seen that without exception more training data results in higher overall *F*-scores except at 10% where the result seems to be biased by the small sample, perhaps because one abstract is partly included in the training and testing sets. As we would expect larger training sets reduce the effects of data sparseness and allow more accurate models to be induced.

The rate of increase in improvement however is not uniform according to the feature sets that are used. For surface word features and head noun features the improvement in performance is consistently increasing whereas the improvement for using orthographic and part of speech features is quite erratic. This may be an effect of the small sample of training data that we used and we could not find any consistent explanation why this occurred.

Looking at a class by class break down of the results allows us to see the influence of features on specific classes in the data set. Results are summarized for 100% data in Table 5. Comparing the F-measures between classes and the class frequencies given in Table 1 it is clear that classes with more examples such as *PROTEIN* have higher *F*-scores and classes with fewer examples such as *RNA* have lower *F*-scores—this reflects the usual limit we expect due to data-sparseness.

As we observed before, the best overall result comes from using Or hd, i.e. surface words, orthographic and head features. However the total score hides the fact that three classes, i.e. SOURCE.mo, SOURCE.mu and SOURCE.ti actually perform worse when using anything but surface word forms. One possible explanation for this is that all of these classes have very small numbers of samples and the effect of adding features may be to blur the distinction between these and other more numerous classes in the model. However it is interesting to note that this does not happen with the RNA class which is also very small.

4.2. Experiment 2: effect of feature sets

The effects of feature sets is of major importance in modelling named entity. In general, we would like to identify only the necessary features that are required and to remove those that do not contribute to an increase in performance. This also saves time in training and testing.

The results from Tables 3 and 4 at 100% training data are summarized in Table 6 and clearly illustrate

Table 3 *F*-scores on Bio1 showing the effects of training set size, feature sets, and context window sizes

Feature set and window size	Percentage of data used in experiment									
	10	20	30	40	50	60	70	80	90	100
Wd −10	58.52	47.30	51.44	52.40	52.37	52.30	51.29	53.24	55.57	56.06
Wd −1 + 1	55.35	48.15	53.91	54.50	56.02	55.30	55.92	58.98	60.28	61.55
Wd −2 + 2	46.87	40.73	47.92	49.64	53.31	53.20	55.01	56.95	59.40	62.04
Wd −3 + 2	46.12	38.55	44.19	47.93	49.50	50.50	51.21	54.76	56.66	60.25
Wd −3 + 3	44.83	35.37	42.67	45.24	46.78	49.10	49.66	54.01	55.59	58.83
Or−10	60.33	55.08	63.49	63.41	64.09	63.04	62.97	62.64	64.59	65.63
Or −1 + 1	65.35	58.69	66.63	68.18	69.20	68.74	69.55	69.32	71.02	72.13
Or −2 + 2	60.84	58.90	66.44	67.17	69.88	68.81	69.68	69.62	71.41	72.12
Or −3 + 2	62.48	59.21	65.64	66.69	67.56	67.25	68.37	68.94	69.92	71.69
Or −3 + 3	59.61	58.65	64.95	65.68	67.11	66.65	67.85	68.84	69.54	71.78
Head −10	58.51	47.10	51.99	52.74	52.44	52.01	53.09	53.79	55.97	57.01
Head −1 + 1	57.50	50.00	55.81	57.88	58.03	57.84	58.81	61.08	62.64	63.93
Head −2 + 2	49.43	45.92	53.40	53.75	57.52	56.94	59.33	61.29	63.36	64.67
Head −3 + 2	46.51	39.42	49.39	49.75	54.54	54.81	56.95	58.13	59.25	61.96
Head −3 + 3	45.79	40.81	47.52	48.11	53.58	53.50	55.95	57.02	59.06	61.52
POS −10	61.62	52.89	61.14	62.04	62.62	61.51	61.05	60.78	62.71	62.63
POS −1 + 1	61.24	57.25	63.83	62.94	65.35	64.82	67.40	66.47	67.43	68.37
POS −2 + 2	57.52	53.11	59.39	59.98	62.86	62.16	63.72	64.17	64.56	66.92
POS −3 + 2	56.81	54.55	56.53	56.26	59.60	59.40	61.42	61.86	63.41	64.90
POS −3 + 3	54.76	53.28	56.79	55.02	57.46	57.66	59.60	59.89	62.39	63.50

Wd: surface word level features; Or: orthographic features; Head: head noun features; POS: part of speech features.

Table 4 *F*-scores on Bio1 showing the effects of training set size, feature sets, and context window sizes

Feature set and window size	Percentage of data used in experiment									
	10	20	30	40	50	60	70	80	90	100
Or hd −10	62.16	57.80	64.31	65.70	65.20	63.84	64.90	64.73	66.46	67.31
Or hd −1 + 1	64.84	60.52	68.42	68.25	68.82	69.34	71.31	71.88	72.60	73.38
Or hd −2 + 2	61.16	61.10	68.06	67.42	69.32	69.62	70.91	71.31	72.31	**74.23**
Or hd −3 + 2	61.54	60.06	65.87	66.33	67.43	68.36	70.28	70.15	70.81	72.95
Or hd −3 + 3	59.68	57.03	64.58	65.76	66.84	67.16	69.07	69.22	70.73	72.12
Or POS −10	61.48	54.04	63.20	63.92	64.11	64.74	63.23	63.62	64.87	66.28
Or POS −1 + 1	64.57	58.89	66.52	66.77	67.83	67.90	69.32	69.07	70.84	71.70
Or POS −2 + 2	61.48	58.56	63.37	65.44	67.01	66.74	68.21	68.55	70.09	71.87
Or POS −3 + 2	61.08	57.14	64.23	63.39	65.53	65.11	67.31	67.78	68.64	71.54
Or POS −3 + 3	57.92	57.12	62.86	62.36	65.48	64.41	66.10	66.64	68.22	70.46
POS hd −10	64.90	55.39	61.14	61.65	61.91	61.29	61.88	60.51	63.27	63.82
POS hd −1 + 1	62.25	57.25	63.66	64.81	64.64	65.57	67.78	67.63	68.69	69.68
POS hd −2 + 2	58.08	53.23	58.91	60.28	62.55	62.06	64.19	64.51	66.18	67.66
POS hd −3 + 2	57.09	53.20	56.58	57.75	59.34	59.14	62.19	62.93	64.23	65.41
POS hd −3 + 3	54.69	51.09	55.67	55.46	58.31	58.28	60.88	61.17	62.94	64.31
Or POS hd −10	63.70	56.63	63.29	65.11	64.72	64.14	64.40	64.04	66.01	67.41
Or POS hd −1 + 1	66.20	59.65	66.49	67.91	68.44	68.14	70.01	70.61	71.80	72.95
Or POS hd −2 + 2	61.62	58.03	64.76	65.16	66.45	67.26	69.00	69.86	70.83	72.56
Or POS hd −3 + 2	62.06	57.28	63.74	64.50	66.10	66.25	68.01	69.05	69.44	71.59
Or POS hd −3 + 3	59.12	56.51	62.43	62.61	65.37	65.09	66.89	67.80	69.36	71.25

Wd: surface word level features; Or: orthographic features; Head: head noun features; POS: part of speech features.

Table 5 Class by class performance using a $-2+2$ window shown against feature sets

NE+ class	Feature set							
	Wd	Or	Head	POS	Or hd	Or POS	POS hd	Or POS hd
DNA	44.53	56.49	50.88	47.33	**62.78**	58.12	47.30	59.19
PROTEIN	65.07	77.50	67.96	72.10	**78.99**	77.03	72.89	77.58
RNA	12.12	42.11	12.90	24.24	**43.24**	37.84	6.67	29.41
SOURCE.cl	52.63	57.14	51.52	54.79	**59.21**	55.90	56.94	59.87
SOURCE.ct	65.83	66.39	66.22	63.70	**69.32**	67.03	65.65	68.94
SOURCE.mo	**32.00**	16.67	9.09	17.39	17.39	16.67	17.39	17.39
SOURCE.mu	**61.02**	58.41	55.24	57.14	51.92	54.55	53.33	51.92
SOURCE.sl	55.22	62.86	62.69	51.20	**68.53**	62.41	54.84	63.38
SOURCE.ti	**23.26**	18.18	0.00	14.63	5.00	14.29	0.00	0.00
SOURCE.vi	76.54	75.16	79.50	73.68	**80.25**	74.84	75.00	73.33

Wd: surface word level features; *O* r: Orthographic features; *Head*: Head noun features; *P* OS: part of speech features.

the value of surface word level features combined with orthographic and head noun features. Orthographic features allow us to capture many generalities that are not obvious at the surface word level such as *IkappaB alpha* and *IkappaB beta* both being PROTEINs and *IL-10* and *IL-2* both being PROTEINs.

The orthographic-head noun feature combination (Or hd) gives the best overall performance of 74.23 at 100% training data on a $-2+2$ window. Overall orthographic features combined with surface word features gave an improvement of between 4.9 and 22.0 per cent. at 100% data depending on window size over surface words alone. This was the biggest contribution by any feature except the surface words. Head information, for example, allowed us to correctly capture the fact that in the phrase *N* F-kappaB consensus site the whole of it is a DNA, whereas using orthographic information alone the SVM could only say that *N* F-kappaB was a PROTEIN and ignoring *consensus site*. We see a similar case in the phrase *primary NK cells* which is correctly classified as SOURCE.ct using head noun and orthographic features but only *NK cells* are found using orthographic features. This mistake is a natural consequence of a limited contextual view which the head noun feature helped to rectify.

Part of speech (POS) when combined with surface word features gave an improvement of between 7.9 and 11.7% at 100% data. The influence of POS though does not appear to be sustained when combined with other features and we found that it actually degraded performance slightly in many cases. This may possibly be due to either overlapping knowledge or more likely subtle inconsistencies between POS features and say, orthographic features. This could have occurred during training when the POS tagger was trained on an out of domain (news) text collection. It is possible that if the POS tagger was trained on in-domain texts it would make a greater and more consistent contribution. An example where orthographic features allowed correct classification but adding POS features resulted in failure is *p50* in the phrase *consisting of 50 (p50)- and 65 (p65)-kDa proteins*. Also in the phrase *c-Jun transactivation domain* where only *c-Jun* should be tagged as a protein, by using orthographic features and POS the model tags the whole phrase as a PROTEIN. This is probably because POS tagging gives a NN feature value (common noun) to each word. This is very general and does not allow the model to discriminate between them.

The fourth feature we investigated is related to syntactic rather than lexical knowledge. We felt though that there should exist a strong semantic relation between a word in a term and the head noun of that term. The results in Table 6 show that

Table 6 Summary of the interaction between window size and feature sets

Window size	Feature set							
	Wd	Or	Head	POS	Or hd	Or POS	POS hd	OrPOS hd
-10	56.06	65.63	57.01	62.63	67.31	66.28	63.82	67.41
$-1+1$	61.55	72.13	63.93	68.37	73.38	71.70	69.68	72.95
$-2+2$	62.04	72.12	64.67	66.92	**74.23**	71.87	67.66	72.56
$-3+2$	60.25	71.69	61.96	64.90	72.95	71.54	65.41	71.59
$-3+3$	58.83	71.78	61.52	63.50	72.12	70.46	64.31	71.25

Wd: surface word level features; Or: Orthographic features; Head: head noun features; POS: part of speech features.

Table 7 Accuracy of term boundary identification given according to the feature set and window size

Feature set	Window size				
	−10	−1 + 1	−2 + 2	−3 + 2	−3 + 3
Only words	63.80	68.71	69.14	68.74	67.91
Orthographic	78.10	79.24	79.68	79.05	78.97
Head	66.22	70.68	72.64	71.60	71.02
POS	75.27	75.71	75.08	73.93	73.32
Orth and head	79.48	79.99	**80.46**	79.78	79.39
Orth and POS	78.86	79.66	79.24	78.29	78.99
POS and head	75.26	76.40	76.00	74.53	74.38
Orth, POS and head	79.33	79.75	79.90	79.53	79.06

Wd: surface word level features; Or: orthographic features; Head: head noun features; POS: part of speech features.

while the overall contribution of the *Head* feature is quite small, it is consistent and significant ranging from about 1.7—4.6% *F*-score improvement depending on window size at 100% training data. Moreover, *Head* features contribute to improving the overall *F*-score when combined with surface words and orthographic features.

4.3. Experiment 3: effect of window size

The influence of the training window size has a significant influence on the final result. Our initial expectation was that a larger training window ($-3+3$) would prove to be the best as the SVM model should be free to decide on the contribution of each part of the window. However from empirical analysis it is clear that the effects of different parts of the window do not give a uniform improvement. For example, in Table 6 we see that comparing $-2+2$ and $-3+2$ leads to a clear drop in performance for nearly all feature sets. By including both the -3 or the +3 features we can expect a drop in performance below what we could obtain in just a small $-1+1$ window.

Unexpectedly, we also found a strong relationship between training set size and window size. At small data sizes a narrow window is better as shown by the underlined results in Table 3. It seems therefore that due to limited training data our model achieved best performance at $-1+1$ but a larger window may achieve better results on larger data sets.

4.4. Experiment 4: effect of boundary identification

The previous experiments considered a combined task of identification and classification. Here, we look at the effect of boundary identification on the result. By ignoring term class and just measuring the ability of the model to find the start and end word tokens for each named entity expression, we obtained the results shown in Table 7. The results reflect many of the previously reported findings, such as a $-2+2$ window being the best and -3 and +3 window features being harmful.

What also seems clear from the boundary identification results is that performance for using surface or head noun features is quite dependent on the window size but for POS or orthographic features there is less correlation. Moreover, while POS features seem beneficial at small window sizes their benefit reduces for larger context windows.

5. Conclusion

The method we have shown for identifying and classifying technical terms has the advantage of being portable, not requiring large domain dependent dictionaries and no hand-made patterns were used. Additionally, since all the word level features are found automatically there is no need for intervention to create domain specific features. Indeed, the only thing that is required is a quite small corpus of text containing entities tagged by a domain expert.

Our experiments on a molecular biology text collection of MEDLINE abstracts have shown a number of underlying factors that should enable better tools to be built in the future. These include:

- the optimal context window size for training may not be the largest one and seems to be directly related to the amount of training data. A $-2+2$ window was found to be best in our tests;
- a combination of surface words, orthographic features and head noun features were found to give the best results among those tested;
- part of speech taggers may need to be trained in-domain to give any benefit to performance;
- the influence of the head noun feature is small but overall beneficial in helping resolve surface term transformations such as coordination.

For future work, we are now looking at how to balance the scores from SVM for each word-class over the whole of a sentence using dynamic programming. Theoretically, the existing SVM model cannot consider evidence from outside the context window, in particular evidence related to named entity class scores in the history and later in the sentence. We are also now developing a larger annotated test collection of 100 *EMBO Journal* full papers using the same top level ontology as used in these experiments. This will provide a far more challenging and realistic collection on which to test our models while hopefully maintaining the high quality levels of the original Bio1 test collection which we regard as a key point in constructing accurate NE classifiers.

Acknowledgements

This work was supported in part by the Japan Society for the Promotion of Science (grant no. 14701020) and by a Leadership Fund grant from the National Institute of Informatics. The authors would like to thank Jun-ichi Tsujii (University of Tokyo) for providing the data set Bio1. We would also like to thank the anonymous reviewers for their many helpful comments.

References

[1] Thomas J, Milward D, Ouzounis C, Pulman S, Carroll M. Automatic extraction of protein interactions from scientific abstracts. In: Altman RB, Dunker AK, Hunter L, Klein TE, editors. Pacific Symposium on Biocomputing 5. Honolulu, Hawaii, USA; 2000;538—49.

[2] Craven M, Kumlien J. Constructing biological knowledge bases by extracting information from text sources. In: Lengauer T, Schneider R, Bork P, Brutlag D, Glasgow J, werner Mewes H, Zimmer R, editors. Proceedings of the Seventh International Conference on Intelligent Systemps for Molecular Biology (ISMB-99). 1999.1-57735-083-9. p. 77—86.

[3] de Bessé B, Nkwenti-Azek B, Sager JC. Glossary of terms used in terminology. Terminology 1997;4(1):117—56.

[4] Justeson J, Katz S. Technical terminology: some linguistic properties and an algorithm for identification in text. Nat Lang Eng 1995;28:9—27.

[5] DARPA. Proceedings of the Sixth Message Understanding Conference(MUC-6). Columbia, MD, USA: Morgan Kaufmann; 1995.

[6] Bikel D, Miller S, Schwartz R, Wesichedel R. Nymble: a high-performance learning name-finder. In: Grishman R, editor. Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP'97). 1997. p. 194—201.

[7] Freitag D, McCallum A. Information extraction with HMMs and shrinkage. In: Proceedings of the Workshop on Machine Learning for Information Extraction held at AAAI 1999. 1999.

[8] Sekine S, Grishman R, Shinnou H. A decision tree method for finding and classifying names in Japanese texts. In: Charniak E, editor. Proceedings of the Sixth Workshop on Very Large Corpora (WVLC'98) at COLING-ACL'98, 1998.

[9] Borthwick A, Sterling J, Agichtein E, Grishman R. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In: Charniak E, editor. Proceedings of the Sixth Workshop on Very Large Corpora (WVLC'98) at COLING-ACL'98. 1998. p. 152—60.

[10] Herzig T, Johns M. Extraction of medical information from textual sources: a statistical variant of the boundary word method. In: Masys DR, editor. Proceedings of the American Medical Informatics Association (AMIA) 1997 Annual Fall Symposium, 1997.

[11] Vapnik VN. The nature of statistical learning theory. New York: Springer-Verlag; 1995.

[12] Takeuchi K, Collier N. Use of support vector machines in extended named entity recognition. In: Roth D, van den Bosch A, editors. Proceedings of the Sixth Conference on Natural Language Learning 2002 (CoNLL-2002). 2002. p. 119—25.

[13] Nobata C, Collier N, Tsujii J. Comparison between tagged corpora for the named entity task. In: Kilgarriff A, Sardinha TB, editors. Proceedings of the Workshop on Comparing Corpora at the Association for Computational Linguistics (ACL'2000). 2000. p. 20—7.

[14] MEDLINE. The PubMed database can be found at: http://www.ncbi.nlm.nih.gov/PubMed/; 1999.

[15] Kashyap V, Sheth A. Semantic heterogeneity in global information systems: the role of metadata, context and ontologies. In: Papzoglou M, Schlageter G, editors. Cooperative information systems: current trends and directions. Academic Press; 1996.

[16] Gruber TR. A translation approach to portable ontology specifications. Knowl Acquisit 1993;5(2):199—221.

[17] Lindberg DA, Humphreys L, Betsy T, McCray. Alexa. The unified medical language system. Methods Informat Med 1993;32:281—91.

[18] Lovis C, Michel P, Baud R, Scherrer J. Word segmentation processing: a way to exponentially extend medical dictionaries. Medinfo 1995;8:28—32.

[19] NLM. Medical subject headings, bethesda, MD: National Library of Medicine; 1997.

[20] Bairoch A, Apweiler R. The SWISS-PROT protein sequence data bank and its new supplement TrEMBL. Nucleic Acids Res 1997;25:31—6.

[21] Tateishi Y, Ohta T, Collier N, Nobata C, Ibushi K, Tsujii J. Building an annotated corpus in the molecular-biology domain. In: Workshop on Semantic Annotation and Intelligent Content. Luxemburg; 2000 (held in conjunction with COLING'2000).

[22] Collier N, Takeuchi K, Kawazoe A, Mullen T, Wattarujeekrit T. A framework for integrating deep and shallow semantic structures in text mining. In: Palade V, Howlett R, Jain L, editors. Proceedings of the Seventh International Conference on Knowledge-based Intelligent Information and Engineering Systems (KES'2003). Lecture Notes in Computer Science, vol. 2773. Springer-Verlag; 20033-540-40803-7.

[23] Burges C. A tutorial on support vector machines for pattern recognition. Data Mining Knowl Discov 1998;2(2): 121—67.

[24] Cristianini N, Shawe-Taylor J. An introduction to support vector machines and other kernel-based learning methods. Cambridge, England: Cambridge University Press; 2000, 0521780195.

[25] Trends and controversies: support vector machines. IEEE Intell Sys 1998:18—28.
[26] Vapnik VN. Estimation of dependencies basd on empirical data. New York: Springer-Verlag; 1982.
[27] Schölkopf B, Smola A, Müller K, Burges C, Vapnik V. Support vector methods in learning and feature extraction. Aust J Intell Inform Process Sys 1998;1:3—9.
[28] Chapelle O, Vapnik V, Bousquet O, Mukherjee S. Choosing multiple parameters for support vector machines. Machine Learning 2002;46(1):131—59.
[29] Cortes C, Vapnik V. Support-vector networks. Machine Learning 1995;20:273—97.
[30] Joachims T. Making large-scale SVM learning practical. In: Scholkopf B, Burges C, Smola A, editors. Advances in Kernel methods—support vector learning. Cambridge, MA: MIT Press; 1999. p. 169—84.
[31] Kreßel U. Pairwise classification and support vector machines. In: Scholkopf B, Burges C, Smola A, editors. Advances in Kernel methods—support vector learning. Cambridge, MA: MIT Press; 1999. p. 255—68.
[32] Brill E. A simple rule-based part of speech tagger. In: Third Conference on Applied Natural Language Processing (ANLP'92)—Association for Computational Linguistics. Cambridge, MA: The MIT Press; 1992. p. 152—5.
[33] Tapanainen P, Järvinen T. A non-projective dependency parser. In: Grishman R, editors. Proceedings of the Fifth Conference on Applied Natural Language Processing. Washington Marriot Hotel, Washington, DC: Association of Computational Linguistics; 1997. p. 64—71.
[34] van Rijsbergen CJ. Information retrieval. London: Butterworths; 1979.