行政院國家科學委員會專題研究計畫　成果報告

自然語言處理技術在生物資訊自動粹取上的研究(2/2)

計畫類別：個別型計畫
計畫編號：NSC92-2213-E-002-022-
執行期間：92 年 08 月 01 日至 93 年 07 月 31 日
執行單位：國立臺灣大學資訊工程學系暨研究所

計畫主持人：陳信希

報告類型：完整報告

處理方式：本計畫可公開查詢

中　華　民　國 93 年 10 月 11 日

行政院國家科學委員會補助專題研究計畫成果報告

自然語言處理技術在生物資訊自動粹取上的研究

計畫類別：■ 個別型計畫　　□ 整合型計畫
計畫編號：NSC 92－2213－E－002－022
執行期間：92 年 8 月 1 日至 93 年 7 月 31 日

計畫主持人：陳信希教授
共同主持人：
計畫參與人員：侯文娟、李遲

成果報告類型(依經費核定清單規定繳交)：□精簡報告　■完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立臺灣大學資訊工程學系

中華民國九十三年十月八日

Abstract

Named entity (NE) recognition is a fundamental task in biomedical data mining. Multiple-class annotation is more challenging than single-class annotation. Moreover, extracting newly discovered functional features from the massive literature is other major challenging issue. In this project, we first we focused on the experiments of protein/gene names. We considered protein/gene collocates extracted from biological corpora as restrictions to enhance the precision rate of protein/gene name recognition. In addition, we integrate the results of multiple NE recognizers to improve the recall rates. Then, we tried the extension to recognize the multiple-class named entities. In this study, we took a single word classification approach to deal with the multiple-class annotation problem using Support Vector Machines (SVMs). Finally, we considered the issues of biological relationship mining, and had an experiment on automatically annotating the Gene References into Function (GeneRIF) in a new literature. In this project, when we focused on the enhancement of performance of protein and gene name recognizers, Yapex and KeX, and ABGene and Idgene are taken as examples of protein and gene name recognizers, respectively. The precision of Yapex increases from 70.90% to 85.84% at the low expense of the recall rate (i.e. it only decreases 2.44%) when collocates are incorporated. When both filtering and integration strategies are employed together, the Yapex-based integration with KeX shows good performance, i.e., the F-score increases by 7.83% compared to the pure Yapex method. The results of gene recognition show the same tendency. The ABGene-based integration with Idgene shows a 10.18% F-score increase compared to the pure ABGene method. These successful methodologies can be easily extended to other name finders in biological documents. For the automatically multiple-class classification, word attributes, results of existing gene/protein name taggers, context, and other information are important features for classification. For the experiments on GRIF extraction, we tried to find GRIF words in a training corpus, and then applied these informative words to annotate the GeneRIFs in abstracts with several different weighting schemes. The experiments showed that the Classic Dice score is at most 50.18%. In contrast, after employing Support Vector Machines (SVMs) and definition of classes proposed by Jelier *et al*. (2003), the score greatly improved to 56.86% for Classic Dice (CD). Adopting the same features, SVMs demonstrated advantage over the Naïve Bayes Classifier.

Keywords: named entity recognition; biological collocates; collocation model; *t*-test; Gene References into Function; Support Vector Machines

摘要

在生物醫學的資料探勘上，具名實體辨識是最基礎的工作，而多種類的辨識則比單一具名實體辨識更加有挑戰性。此外，如何從大量文件中擷取新的功能特徵，也是另一項十分有挑戰性的議題。在本篇報告中，我們首先針對蛋白質及基因進行增加辨識效能的實驗。我們利用從生物語料庫擷取出來的搭配詞語增加蛋白質/基因辨識的準確率，並且整合多個具名實體辨識器的結果以便增加召回率。接下來，我們更擴展辨識範圍至多種類實體辨識，在本篇報告中，我們使用單一字詞分類的方法處理此問題，並且使用支援向量機進行訓練及學習。最後，我們探討生物關係的探勘，並且提出方法，以便自動從文件中抽取出基因功能。在本篇報告中，為了增加蛋白質及基因辨識效能時，我們利用 Yapex 、KeX、ABGene 及 Idgene 進行整合，當使用搭配詞語篩選時，Yapex 的準確率可從 70.90 個百分比上升至 85.84 個百分比，而召回率只下降 2.44 個百分比。若同時使用篩選及整合策略，以 Yapex 為主的整合方法會得到較佳的結果，也就是 F 分數可提升 7.83 個百分比；有關基因辨識的結果也顯示相同趨勢。接下來，在多種類實體辨識實驗中，字的屬性、蛋白質/基因辨識器辨識結果、本文上下關係及其他資訊都會當做分類時的特徵。另外，在基因功能的擷取上，我們首先從訓練語料中找到 GRIF 字詞，再配合各種加權方式以便標記出基因功能，本實驗最好可達的 classic dice 分數為 50.18 個百分比、若再使用支援向量機進行學習，則可提升至 56.86 個百分比。此外，我們也證明使用相同特徵，支援向量機的表現會優於 Naïve Bayes 分類器。

關鍵字：具名實體辨識、生物搭配詞語、詞語同現模型、t 試驗、基因功能、支援向量機。

Index

1. Introduction

The volume of on-line material in the biomedical field has been growing steadily for more than 20 years. Several attempts have been made to mine knowledge from biomedical documents, such as identifying gene/protein names, recognizing protein interactions, and capturing specific relations in databases. Among these, named entities are basic constituents in a document and recognizing them is a fundamental step for document understanding. In the message understanding competition MUC (DARPA, 1998), named entity extraction was one of the evaluation tasks. The named entities included organizations, people, locations, date/time expressions, monetary expressions and percentage expressions. Several approaches have been proposed to capture these types of terms. For example, corpus-based methods are employed to extract Chinese personal names, and rule-based methods are used to extract Chinese date/time expressions as well as monetary and percentage expressions (Chen and Lee, 1996; Chen, et al. 1998). The corpus-based approach is adopted because a large personal name database is available for training. In contrast, rules that have good coverage exist for date/time expressions, so that the rule-based approach is adopted.

In the past, named entity extraction has mainly focused on general domains. However, many scientific documents have been published recently, especially in the biomedical domain. Several attempts have been made to mine knowledge from biomedical documents (Hirschman et al., 2002), such as identifying protein/gene names, recognizing protein interactions, and capturing specific relations in databases. One of the goals is to construct a knowledge base automatically and find new information embedded in documents (Craven and Kumlien, 1999). Craven and Kumlien (1999) identified that the information extraction task may include sub-cellular/cell localization of proteins, tissue localization of proteins, and drug interactions with a given protein. Similar information extraction works have been explored in this domain. Named entities, such as protein names, gene names, drug names, disease names, and so on, have also been recognized (Collier et al., 2000; Fukuda et al., 1998; Hanisch et al., 2003; Krauthammer et al., 2000; Morgan et al., 2003; Olsson et al., 2002; Rindflesch et al., 2000; Tanabe and Wilbur, 2002; Yamamoto et al. 2003). Some of them have used machine learning methods, e.g., Hidden Markov Models (HMMs), and Support Vector Machines (SVMs), to recognize protein/gene names (Collier et al., 2000; Hanisch et al., 2003; Morgan et al., 2003; Rindflesch et al., 2000; Tanabe and Wilbur, 2002; Yamamoto et al. 2003). Others have used knowledge-based rules, accompanied by lexical or morphological analysis, to help with protein/gene name detection (Fukuda et al., 1998; Krauthammer et al., 2000; Olsson et al., 2002). The relationships between these entities, e.g., protein-protein, gene-gene, drug-gene, drug-disease, etc., have also been extracted (Adamic et al., 2002; Blaschke et al., 1999; Friedman et al., 2001; Hou and Chen, 2002; Hou and Chen, 2003; Marcotte et al., 2001; Ng and Wong, 1999; Park et al., 2001; Rindflesch et al., 2000; Thomas et al., 2000; Tsuruoka and Tsujii, 2003; Wong, 2001). EDGAR (Rindflesch et al., 2000) used a POS tagger, NLP techniques, other knowledge sources and contextual rules to identify the relationships between genes and drugs in cancer therapy. Meanwhile, Adamic, *et al*. (2002) used a statistical method to identify gene-disease connections from literature. Protein/gene interactions have been discovered automatically in the literature by methods that utilized natural language processing, parsing techniques or the analysis of sentences that discussed interactions by using frequency analysis of individual words (Blaschke et al., 1999; Friedman et al., 2001; Marcotte et al., 2001; Ng and Wong, 1999; Park et al., 2001; Rindflesch et al., 2000; Tsuruoka and Tsujii, 2003). Other relationships were extracted to improve the performance of named entity recognition, e.g., through the information supplied

from protein/gene keywords (Hou and Chen, 2002; Hou and Chen, 2003) or the Naïve Bayes classifier (Tsuruoka and Tsujii, 2003).

Named entity recognition is a fundamental step for mining knowledge from biological articles. After identifying named entities, most research (Blaschke et al., 1999; Ng and Wong, 1999; Park et al., 2001; Rindflesch et al., 2000; Sekimizu et al., 1998) has been based on some special verbs and their related noun forms to discover molecular pathways or relationships. These pre-specified words indicate actions associated with protein or gene interactions. Blaschke, *et al.* (1999) used fourteen keywords for protein-protein interactions from MEDLINE articles. Ng, *et al.* (1999) applied some function words for the *inhibit-activate* relationships. Sekimizu, *et al.* (1998) extracted gene relations associated with seven frequently used verbs found in MEDLINE abstracts. In all these papers, with the exception of Sekimizu, the keywords are listed by intuition. Some keywords are common to most of the papers, while some are special. The problem with the above approaches is that we cannot be sure if the keyword set is complete for mining biological relationships. This motivated us to find biological keywords in an automatic way.

Collocation denotes two or more words that have strong relationships (Manning and Schutze, 1999). For example, if the phrase "NF-kappa B activation" often appears in a sentence where "NF-kappa B" is a protein name, it means that "NF-kappa B" and "activation" are collocations, i.e., "NF-kappa B" and "activation" occur together in the document. The related technologies have been applied to terminological extraction, natural language generation, parsing, and so on. This paper deals with two special collocations in the biological domain – namely: protein collocation and gene collocation. We will determine those keywords that co-occur with protein or gene names by using statistical methods. Such terms, referred to as *collocates* of proteins or genes hereafter, will be considered as restrictions in protein/gene name extraction. In the former example of "NF-kappa B activation", "activation" is the collocate of the protein "NF-kappa B". Improving the precision rate, without substantially lowering the recall rate is the primary goal of this approach. Furthermore, how to improve the recall rate at a small expense to the precision rate is another interesting topic. We will explore this issue by introducing an integration of multiple name recognizers. In summary, the first motivation of this project is to increase the performance of existing molecular name detectors. The methods we adopted will be explained in Sections 3 - 5.

Previous approaches on biological named entity extraction can be classified into two types – rule-based (Fukuda *et al.*, 1998; Olsson *et al.*, 2002; Tanabe and Wilbur, 2002) and corpus-based (Collier *et al.*, 2000; Chang *et al.*, 2004). Yapex (Olsson *et al.*, 2002) implemented some heuristic steps described by Fukuda, *et al.*, and applied filters and knowledge bases to remove false alarms. Syntactic information obtained from the parser was incorporated as well. GAPSCORE (Chang *et al.*, 2004) scored words on the basis of statistical models that quantified their appearance, morphology and context. The models include Naive Bayes (Manning and Schutze, 1999), Maximum Entropy (Ratnaparkhi, 1998) and Support Vector Machines (Burges, 1998). GAPSCORE also used Brill's tagger (Brill, 1994) to get the POS tag to filter out some words that are clearly not gene or protein names. Efforts have been made (Hou and Chen, 2002, 2003; Tsuruoka and Tsujii, 2003) to improve the performance. The nature of classification makes it possible to integrate existing approaches by extracting good features from them. Several works employing SVM classifier have been done (Kazama *et al.*, 2002; Lee *et al.*, 2003; Takeuchi and Collier, 2003; Yamamoto *et al.*, 2003), and will be discussed further in the rest of this report. In this report, we addressed the task of recognizing biological named entities as a multi-class classification problem with SVMs and extended the idea of collocation to generate features at word and

pattern level in our method. Existing protein/gene recognizers were used to perform feature extraction as well.

Text Retrieval Conference (TREC) has been dedicated to information retrieval and information extraction for years. TREC 2003 introduced a new track called Genomics Track (Hersh and Bhupatiraju, 2003) to address the information retrieval and information extraction issues in the biomedical domain. For the information extraction part, the goal was to automatically reproduce the Gene Reference into Function (GeneRIF) resource in the LocusLink database (Pruitt *et al.*, 2000). GeneRIF associated with a gene is a sentence describing the function of that gene, and is currently manually generated. Consequently, we made the experiments of biological domain on the information extraction task (i.e., secondary task). The goal of this task is to reproduce the GeneRIF annotation from an article. Bhalotia *et al.* (2003) converted this task into a binary classification problem and trained a Naïve Bayes classifier with kernels. The title and last sentence of an abstract were concatenated and features were then extracted from the resulting string. Jelier *et al*. (2003) observed the distribution of target GeneRIFs in 9 sentence positions and converted this task into a 9-class classification problem. Both works indicated that the sentence position is of great importance. We therefore modified our system to incorporate the position information with the help of SVMs and we also investigated the capability of SVMs versus Naïve Bayes on this problem.

The rest of this paper is organized as follows. The protein and the gene name recognizers used in this study are introduced in Section 2. The collocation method we adopted is described in Section 3. The filtering and the integration strategies are proposed in Sections 4 and 5, respectively and the experimental results of these two strategies are shown and discussed. The methods for annotating multiple types of biological entities and results are presented in Section 6. Section 7 presents the architecture, experimental methods and results for the extraction of gene functions. Finally, in Section 8, we present our conclusions and suggest the direction of future research.

2. Molecular Name Recognizers

The detection of molecular names such as proteins and genes presents a challenging task due to their variant structural characteristics, their resemblance to regular noun phrases and their similarity to other kinds of biological substances. Many irregularities and ambiguities exist in gene and protein nomenclature. For example, protein/gene names may be synonymous with common words, such as “ran”, “envelope”, “cat”, etc. In addition, some principles of the nomenclature are similar to chemicals, e.g., “Ca2+-ATPase” is a protein while “Ca2+” is a chemical. Consequently, several issues have to be addressed during protein/gene name recognition.

Previous approaches to biological named entity extraction can be classified in two types – namely: rule-based (Fan, 2003; Fukuda et al., 1998; Humphreys et al., 2000; Olsson et al., 2002; Tanabe and Wilbur, 2002) and corpus-based (Collier *et al*., 2000; Chang *et al.*, 2004). KeX developed by Fukuda, *et al*. (1998) and Yapex developed by Olsson, *et al*. (2002) were based on handcrafted rules for extracting protein names. Kex used surface clues like upper case letters, numerical letters and symbols to extract core terms and later connected them to other terms in the surrounding text (Fukuda et al., 1998). Yapex first implemented some heuristic steps described by Fukuda, *et al*. (1998), and then applied filters and knowledge bases to remove false hits. Finally, Yapex utilized the syntactic information from the parser to identify protein names.

ABGene developed by Tanabe, *et al*. (2002) used Brill’s tagger (Brill, 1994) as the fundamental extraction program, followed by additional layers of post-processing rules to filter out false positives, as well as to recover false negatives in the first-step tagging of gene

and protein names. Brill's tagger assigns part-of-speech tags to words. For example, for the title "Genetic characterization in two Chinese women", Brill's tagger will produce the tagged result "Genetic/JJ characterization/NN in/IN two/CD Chinese/JJ women/NNS" to indicate "Genetic" as an adjective, "characterization" as a common noun, "in" as a preposition, "two" as a cardinal number, "Chinese" as an adjective and "women" as a plural common noun. Since gene names are usually single nouns or noun phrases, it is helpful to recognize gene names by applying Brill's tagger. After tagging, the post-processing rules are used to filter out false positives and recover false negatives. For filtering false positives, on the one hand, ABGene precompiles some general biological terms (acids, antigen, etc.), amino acid names, restriction enzymes, cell lines and organism names. On the other hand, ABGene uses regular expressions to indicate that a word is not a gene name, e.g., common drug suffixes (-ole, -ane, -ate, etc.). For recovering false negatives, ABGene applies contextual rules to find compound names. For example, one rule is "ANYGENE CC x", where "ANYGENE" is a tagged gene, "CC" is a coordinating conjunction and "x" is the current word. The constraint of this rule is that "x" contains a capital letter, dash or number, and is not a verb or an adverb. If matched, the tag of "x" will be changed to CONTEXTGENE. Finally, compound names are found. Some examples of filtering out false positives and recovering false negatives are described in (Tanabe and Wilbur, 2002). Idgene developed by Fan (2003) is a dictionary-based gene name identification program. The basic idea of Idgene is to use exact match for gene symbols and fuzzy match for gene names/phenotypes, which gives a suggestion list of the hit genes weighted by surrounding contexts. Idgene also uses Brill's tagger to get POS tags, and then computes the scores of the exact/fuzzy matches. Finally, Idgene merges shorter terms with longer terms to obtain the final scores. Both ABGene and Idgene utilize some hand-made rules for extracting gene names. Collier, *et al*. (2000) adopted a machine learning approach that involved training a Hidden Markov Model with a small corpus of 100 MEDLINE abstracts to extract the names of gene and gene products.

Different taggers have their own specific features. Idgene was evaluated on 156 Chinese Gene Variation papers selected from 1997-1998 BIOSIS Previews and EMBASE (BIOSIS organization, 1999). It had a 24.68% precision rate and an 85.39% recall rate. ABGene was developed as a general-purpose gene tagger. Fan (2003) applied ABGene to the same test collection as the one used in Idgene. ABGene achieved a 31.32% precision rate and an 81.46% recall rate. KeX was evaluated by using 30 abstracts of SH3 domain and 50 abstracts of signal transduction. It achieved a 94.70% precision rate and a 98.84% recall rate. Yapex was applied to a test corpus of 101 MEDLINE abstracts. Of these, 48 documents were obtained from queries about protein binding and interaction, and 53 documents were randomly chosen from the GENIA corpus (Collier et al., 1999). The query posed to MEDLINE was "protein binding [Mesh term] AND interaction AND molecular" with the parameters *abstract, English, human, publication date 1996-2001*. The performance of tagging protein names was 70.90% for precision and 69.53% for recall. When the same test corpus was applied to KeX, it achieved a 40.41% precision rate and a 41.13% recall rate. These results show that each tagger has its own characteristics and changing the domain may result in the variant performances. Therefore, how to select the correct molecular entities proposed by the existing taggers is an interesting issue.

3. Statistical Methods for Collocation

The overall flow of our method is shown in Figure 1. To extract protein/gene collocates, we need a corpus in which protein/gene names have been tagged. Preparing a tagged biological corpus is the first step, after which common stop words are removed and stemming (e.g., map "listed" and "listing" to its root form "list") is applied to gather and group more informative

words. The collocation values of the proteins/genes and surrounding words are then calculated. Finally, these values are employed to determine which neighbouring words are the desired collocates. The major modules are specified in detail in the following subsections.

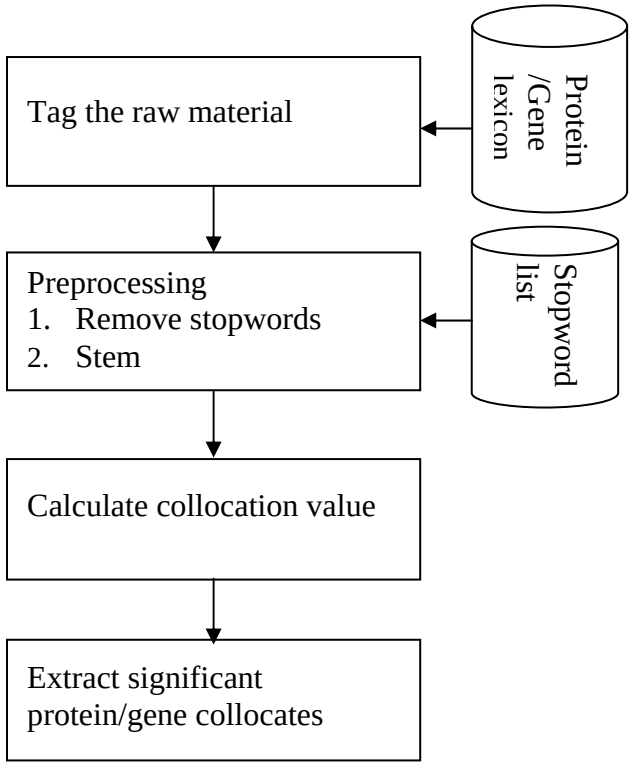

Figure 1. Flow of Mining Protein/Gene Collocates

3.1 Step 1: Tagging the Corpus

In order to calculate the collocation values of words with proteins/genes from a corpus, it is first necessary to recognize protein/gene names. Nevertheless, the goal of this paper deals with the performance issue of protein/gene name tagging. Hence, preparing a protein/gene name tagged corpus and developing high performance protein/gene name taggers seem to be a chicken-and-egg problem. Because the corpus developed in the first step is used to extract the contextual information of proteins/genes, a completely tagged corpus is not necessary at the first step. A dictionary-based approach for molecular name tagging, i.e. full pattern matching between the dictionary entries and the words in the corpus, is simple. The major problem is its coverage. Those protein/gene names that are not listed in the dictionary, but appear in the corpus will not be recognized. Thus, this approach only produces a partially tagged corpus, which is sufficient to acquire contextual information for use later in this research.

3.2 Step 2: Preprocessing

3.2.1 Step 2.1: Exclusion of Stopwords

Stopwords are common English words (such as the preposition “in” and the article “the”) that frequently appear in the text, but are not helpful in discriminating special classes. Because they are distributed throughout the corpus, they should be filtered out to remove their

unnecessary impact in the text. The stopword list in this study was collected with reference to the stoplists of Fox (1992), but words that also appeared in the protein/gene lexicon have been removed from the stoplist. For example, "of" is a constituent of the gene name "translocase of inner mitochondrial membrane 8 homolog A", so "of" is excluded from the stoplist. The major reason for excluding such stopwords from the Fox list is to enable exact pattern matching with protein/gene names. Finally, 387 stopwords were used.

3.2.2 Step 2.2: Stemming

Stemming is the procedure of transforming a word from an inflected form to its root form. For example, "suggested" and "suggestion" will be mapped into the root form "suggest" after stemming. The procedure can group the words with the same semantics and therefore reflect more information around the proteins/genes.

3.3 Step 3: Computing Collocation Statistics

Pearson (2001) has discussed problems of gene nomenclature in detail. The irregularity and the ambiguities in gene and protein nomenclature make name identification more difficult. From one MEDLINE abstract, we have the following title: "The relationship between Ca2+-ATPase and freely exchangeable Ca2+ in the dense tubules: a study in platelets from women." In this example "Ca2+-ATPase" is a protein, while "Ca2+" is a chemical. However, they are both composed of letters plus numbers and symbols. Obviously, the nomenclature rules are irregular, so we must find other clues to help name recognition. The clues here are in the context. For "Ca2+-ATPase", the context is "The relationship between" and "and freely exchangeable Ca2+ in" if we take the three words before, and the five words after it. For "Ca2+", its context is "between Ca2+-ATPase and freely exchangeable" and "in the dense tubules: a". If we know the protein collocates contain "relationship", we can pick "Ca2+-ATPase" as a protein and discard "Ca2+". In such a way, a collocate of protein/gene can help to improve precision. This section proposes three collocation statistics to find the collocates of proteins/genes, which often co-occur with protein/gene names in the corpora.

Frequency

The first statistical method we used in this study was frequency. In this phase, the collocates were selected by frequency. To gather more flexible relationships, we defined a collocation window that has five words on each side of protein/gene names. Then, collocation bigrams at a distance were captured. In general, more occurrences in the collocation windows are preferred, but the standard criteria for frequencies are not acknowledged. For example, "go" occurs in the protein collocation window fourteen times, and "pathway" occurs in the gene collocation windows nine times. How to decide if "go" is a good protein collocate, while "pathway" is not a good gene collocate is a difficult issue. Hence, other collocation models are also considered.

Mean and Variance

The second statistical method we applied was mean and variance. The mean value of collocations can indicate how far collocates are typically located from protein/gene names. Furthermore, this method shows the deviation from the mean. The standard deviation of value zero indicates that the collocates and the protein/gene names always occur at exactly

the same distance equal to the mean value. If the standard deviation is low, two words usually occur at about the same distance, i.e., near the mean value. If the standard deviation is high, then the collocates and the protein/gene names usually occur at random distances.

We used the following formulas to calculate mean and standard deviations, respectively.

$$\bar{d}_i = \frac{\sum_{j=1}^{n\_count_i} d_{ij}}{n\_count_i}$$

$$s_i = \sqrt{\frac{\sum_{j=1}^{n\_count_i} (d_{ij} - \bar{d}_i)^2}{n\_count_i - 1}}$$

Where $\bar{d}_i$ is the average distance for word *i* in the collocation windows. d_{ij} is the distance of the *j*-th occurrence of word *i* away from proteins/genes in the collocation windows. For example, d_{ij}=-1 means the *j*-th occurrence of word *i* is located directly to the left of the proteins/genes in the collocation window. n_count_i is the total number of occurrences of word *i* in the document set. s_i is the standard deviation of d_{ij}.

The following examples illustrate the meaning of mean and variance for the word "activation" and proteins.

(1) *IL-2 gene expression and <prot>NF-kappa B</prot> activation through <prot>CD28</prot> requires reactive oxygen production by <prot>5-lipoxygenase</prot>.*

(2) *Activation of the <prot>CD28 surface receptor</prot> provides a major costimulatory signal for T cell activation.*

In Sentence (1), "*activation*" occurs directly on the right of "*NF-kappa B*" and on the left 2nd position away from "*CD28*". In Sentence (2), "*activation*" occurs on the left 3rd position away from "*CD28 surface receptor*". Thus, the average distance for activation is (1+(-2)+(-3))/3. The result is –1.33, and the standard deviation is $\sqrt{\frac{(1-(-1.33))^2 + (-2-(-1.33))^2 + (-3-(-1.33))^2}{3-1}}$. The value of the standard deviation is equal to 1.472 which means that "*activation*" may occur on the left or right at a distance of 1.472 words away from the average distance, which is -1.33 in this example.

t-test Model

When the values of mean and variance are computed, it is necessary to know that two words do not co-occur by chance. We also need to know if the standard deviation is low enough. In other words, we have to set a threshold in the above approach. To achieve the statistical confidence that two words have a collocation relationship, a hypothesis testing, *t*-test, is adopted.

Consider a document set with total *n* words. The *t*-value for each word *i*, t_i, is formulated as follows:

$$t_i = \frac{\bar{x}_i - u_i}{\sqrt{s_i^2 / N}},$$

where

N = total word frequencies in the window,

$$\bar{x}_i = \frac{n\_count_i}{N},$$

$s_i^2 = p_i \times (1 - p_i)$,

$p_i = n\_count_i / n$,

$u_i = p_{protein/gene} \times p_i$, and

$p_{protein/gene}$ is the probability of protein/gene.

The confidence level, i.e. α, is a statistical calculation that measures the degree of certainty (or likelihood) of a correlation, result or forecast. When α is equal to 0.005, the value of *t* is 2.576. In the *t*-test model, if the *t*-value is larger than 2.576, the word is regarded as a good collocate of a protein/gene with 99.5% confidence.

3.4 Step 4: Extraction of Collocates

MEDLINE is a massive biomedical corpus for information retrieval, information extraction and knowledge discovery. Biomedical experts often explore new developments in special topics by retrieving relevant documents from MEDLINE. To preserve the independence between proteins and genes, we used different document sets as training corpora for proteins and genes in this extraction phase.

In the experiments for proteins, the documents used in TREC 2003 Genome Track (http://medir.ohsu.edu/~genomics/) were considered as the training corpus. The text collection consists of 525,936 MEDLINE abstracts where indexing was completed between 4/1/2002 and 4/1/2003. We applied the procedures Steps 1-3 mentioned in this section to this data collection. There are 57,307 protein collocations generated in Step 3. The collocates are not filtered out by part of speech, so the output may contain nouns, prepositions, numbers, verbs, etc.

In the experiments for genes, the documents gathered from the LocusLink database (Pruitt et al., 2000) (http://www.ncbi.nlm.nih.gov/LocusLink) were adopted as the training corpus. The text collection consists of 30,936 MEDLINE abstracts. Applying Steps 1 - 3 in Section 3 to this document collection, we obtained 14,150 gene collocations.

The collocates extracted from a corpus not only serve as conditions of protein or gene names, but also facilitate the discovery of the relationship between proteins (genes) (Hou and Chen, 2002). Verbs are the major targets in the extraction of biological information, (such as Blaschke, *et al.*,1999; Ng, *et al.*,1999; and Ono, *et al.*, 2001 etc.). This is because the subjects and the objects related to these verbs tend to be names of proteins or genes. To ensure that the collocates selected in Step 3 were verbs, we assigned part of speech to these words. There are 12,826 protein collocates and 3,541 gene collocates. Examples of protein and gene collocates are listed in Tables 1 and 2.

Table 1 Examples of Protein Collocates

bind	active	determine	regulate	involve
refine	resolve	find	express	recognize
inhibit	catalyze	reveal	increase	detect
react	control	study	contain	result

Table 2 Examples of Gene Collocates

active	specify	express	mediate	increase
associate	inhibit	resist	bind	concentrate
regulate	response	study	suggest	stimulate
treat	result	release	depend	decrease

4. Consideration of Precision Rate

4.1 Filtering strategies

For protein/gene name recognition, rule-based systems and dictionary-based systems are usually complementary. Rule-based systems can recognize those protein/gene names not listed in a dictionary, but some false entities may also pass at the same time. For example, both "HCMV" and "NFAT" are composed of capital letters. However, "HCMV" is a virus that may be recognized as protein/gene, whereas "NFAT" is definitely a protein. Other examples are "BL-2", a cell line which may be tagged as a protein/gene name, and "AP-2", which is a protein. Dictionary-based systems can recognize molecular entities in a dictionary, but the coverage of all proteins/genes is a major deficiency. A challenge is how to use dictionary information to correctly identify molecular entities. In this section, we employ collocates of proteins/genes mined earlier to help identify the molecular entities. The Yapex system (Olsson et al., 2002) and ABGene (Tanabe and Wilbur, 2002) are adopted to propose candidates, and protein/gene collocates serve as restrictions to filter out less likely protein/gene names.

The following filtering strategies are proposed. We explain them from a protein viewpoint. Let us assume that the candidate set M0 is the output generated by Yapex.

- M1: For each candidate in M0, we will check if a collocate is found in its collocation window. If it is, we will tag the candidate as a protein name. Otherwise, we will discard it. For example, in the sentence "*IL-2 gene expression and NF-kappa B activation through CD28 requires reactive oxygen production by 5-lipoxygenase.*", Yapex tagged "IL-2", "CD28" and "5-lipoxygenase" as proteins. If "activation" and "reactive" are protein collocates, then "CD28" and "5-lipoxygenase" will be retained, since "activation" and "reactive" occur in the collocation window of "CD28" and "reactive" occurs in the collocation window of "5-lipoxygenase".
- M2: Some of the collocates may be substrings of protein names. We relax the restriction in M1 as follows: If a collocate appears in the candidate, or in the collocation window of the candidate, then we tag the candidate as a protein name; otherwise, we discard it. For example, in the sentence: …, *since FGF-1 -induced Rel/kappaB binding proteins do not contain significant levels of c-Rel and are not identical with the CD28 response complex,* "FGF-1" and "Rel/kappaB binding proteins" are protein names. "FGF-1" can be retained with strategy M1, while "Rel/kappaB binding proteins" cannot because the protein collocate "binding" is located in the window of "FGF-1" and not in the window of "Rel/kappa B binding proteins". If we apply strategy M2, the latter will be found.
- M3: Some protein names may appear more than once in a document. They may not always co-occur with some collocates in each occurrence. In other words, the protein candidate and some collocates may co-occur in the first, the second, or even

the last occurrence. To resolve this problem, we revise M1 and M2 as follows. If there exists a collocate co-occurring with a protein candidate during checking, the candidate without any collocate is kept undecided instead of being discarded. After all the protein names have been examined, those undecided candidates may be considered as protein names if one of their co-occurrences contains any collocate. In other words, as long as a candidate has been confirmed once, it is assumed to be a protein throughout. In this way, there are two filtering alternatives M31 and M32 from M1 and M2, respectively. For example, in the sentence: "*Full activation of the MAP kinases that phosphorylate the Jun activation domain, JNK1 and JNK2, required costimulation of T cells with either TPA and Ca2+ ionophore or antibodies to TCR and CD28.*", there are no protein collocates around "CD28". If we apply strategy M31, "CD28" will be retained as a protein because it has been collocated with protein collocates from other parts of the documents. The example for strategy M32 is the same with the one illustrated for strategy M31. Although there are no protein collocates around proteins "the Jun activation domain" and "CD28", strategy M32 helps recognize them as follows. First, "the Jun activation domain" will be detected because a collocate "activation" appears in the protein name "the Jun activation domain". Furthermore, "CD28" will be retained as a protein because it has been collocated with protein collocates from other parts of the documents.

4.2 Evaluation of filtering strategies

To get an additional objective evaluation, we utilized another corpus of 101 abstracts used by Yapex (http://www.sics.se/humle/projects/prothalt) for protein extraction. Similarly, we used the GENIA corpus version 3.02 (http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA) of 2,000 abstracts for the gene evaluations. Using the test corpora and answer keys provided in the Yapex project and the GENIA project, the evaluation results of filtering strategies for proteins and genes are listed in Tables 3 and 4, respectively. Note that the baseline model M0 was not applied during the filtering strategies.

Table 3. Protein Evaluation on Filtering Strategies

	Precision	Recall	F-score
M0	70.90%	69.53%	70.22%
M1	82.10%	57.42%	69.76%
M2	82.35%	57.96%	70.16%
M31	85.89%	66.48%	76.19%
M32	85.84%	67.09%	76.47%

Table 4. Gene Evaluation on Filtering Strategies

	Precision	Recall	F-score
M0	55.87%	74.56%	65.22%
M1	65.93%	69.50%	67.72%
M2	69.26%	69.89%	69.58%
M31	69.79%	71.46%	70.63%
M32	70.08%	71.89%	70.99%

We can partition the labelled results into four groups:

True Positives (TP): items correctly labelled as positive;

False Positives (FP): items incorrectly labelled as positive;

True Negatives (TN): items correctly labelled as negative;

False Negatives (FN): items incorrectly labelled as negatives.

In Table 3, precision, recall and F-score are calculated according to the following equations:

$$\text{Precision (P)} = \frac{TP}{TP + FP},$$

$$\text{Recall (R)} = \frac{TP}{TP + FN}, \text{ and}$$

$$\text{F-score} = \frac{2PR}{P + R}.$$

Compared with the baseline model M0 in Table 3, the precision rates of all the four models using protein collocates improved more than 11.20%. The recall rates of M1 and M2 decreased 12.11% and 11.57%, respectively. Thus, the overall F-scores of M1 and M2 decreased 0.46% and 0.06%, compared to M0. In contrast, if the decision of tagging was deferred until all the information was considered, the recall rates only decreased by 3.05% and 2.44%, and the F-scores of M31 and M32 increased 5.97% and 6.25% relative to M0. The best strategy, M32, improved the precision rate from 70.90% to 85.84%, and the F-score from 70.22% to 76.47%.

In Table 4, the precision rates of all the four models using gene collocates were improved more than 10.06%. The recall rates of M1 and M2 decreased 5.06% and 4.67%, respectively. Thus, the overall F-scores of M1 and M2 increased 2.50% and 4.36%, compared to M0. If the decision of tagging was deferred until all the information was considered, the recall rates only decreased by 3.10% and 2.67%, and the F-scores of M31 and M32 increased by 5.41% and 5.77% relative to M0. The best one, M32, improved the precision rate from 55.87% to 70.08%, and the F-score from 65.22% to 70.99%. Compared to the experimental results shown in Table 3, the same trends occurred for genes shown in Table 4. The results meet our expectations, i.e., to enhance the precision rate, without significantly reducing the recall rate.

5. Consideration of the Recall Rate

5.1 Integration strategies

Here, we analyze the tagged results from protein/gene taggers. There are four types of errors generated by the taggers.

(1) Type 1: completely wrong labelling, e.g., “HCMV” may be tagged as a protein.

(2) Type 2: partially wrong labelling with some correct components in the tagged results. This is due to a mistake about the boundary. For example, "soluble CD4-IgG" may be tagged as a protein rather than the correct tagging "CD4-IgG".
(3) Type 3: incomplete labelling. For example, "NAFT or AP-1 sites", is an instance of a complete gene, but it may be incompletely labelled as "NAFT or AP-1".
(4) Type 4: missing labelling. This occurs when some protein/gene names are not tagged. For example, "E2F-1" may be considered as a non-protein.

Using the filtering strategies introduced in Section 4.1, the most helpful collocates were of Type 1. For Types 2 and 3, the collocates help a little because they may also appear in the collocation window of the wrong labelled gene/protein names. To solve the errors of Types 2 and 3, there is an additional requirement to determine where the name begins and ends within a sentence. Finally, our filtering method cannot help with Type 4, since we cannot produce untagged names.

In order to improve recall, we introduce integration strategies based on a hybrid concept of two protein/gene name taggers. By employing the integration strategies, we resolve errors of Types 2 and 3 by employing integration strategies. The basic idea is that different protein/gene name taggers have their own specific features such that they can recognize different sets of NEs according to their rules or recognition methods. Among the proposed protein/gene names provided by different systems, there may exist some overlaps and some differences. In other words, a protein/gene name recognizer may tag a protein or gene that another recognizer cannot identify, or both of them may accept certain common molecular entities. The integration strategies are used to select correct protein/gene names proposed by multiple recognizers. In this study, we conducted several experiments for different domains: (1) For protein name recognition, Yapex and KeX are adopted because they are freely available on the web; (2) For gene name recognition, ABGene and Idgene are included because the developers were kind enough to provide the resources for our experiments.

Because protein/gene candidates are proposed by two named entity extractors independently, they may be completely separate, completely the same, overlapped in between, overlapped in the beginning, or overlapped at the end. Figure 2 shows these five cases.

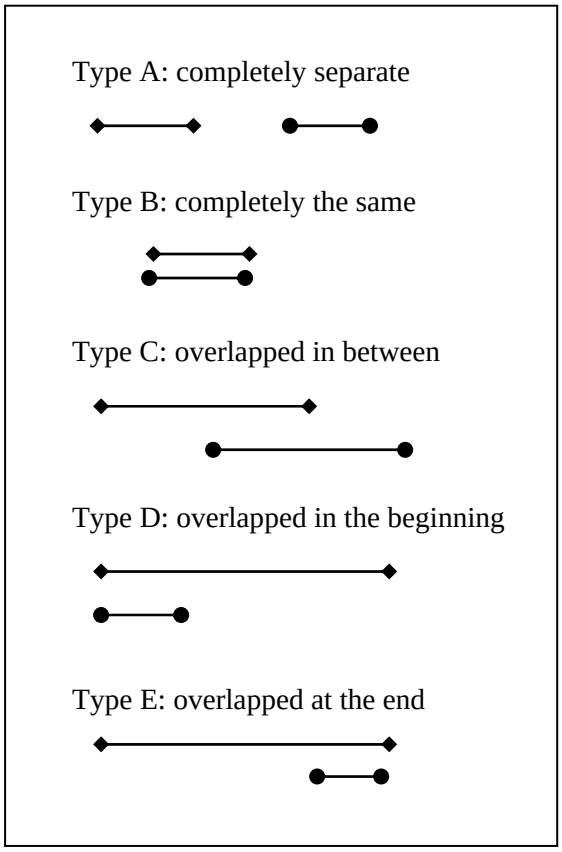

Figure 2. Candidates Proposed by Two Systems

For example, if there is a sentence as follows:

We have previously found a high expression of human Ah receptor (TCDD receptor) mRNA in peripheral blood cells of individuals.

If one system only tagged "*Ah receptor*" as a protein name and the other system proposed "*TCDD receptor*" as a protein name, then this sentence belongs to Type A: completely separate. If two systems all tagged "*TCDD receptor*" as a protein name, this is a case of Type B: completely the same. If one system tagged "*human Ah receptor*" as a protein name, while another tagged "*Ah receptor (TCDD receptor)*" as a protein name, this is a case of Type C: overlapped in between, where "*Ah receptor*" is the overlapped part. If one system tagged "*Ah receptor*" as a protein name and the other one proposed "*Ah receptor (TCDD receptor)*" as a protein name, this is a case of Type D: overlapped in the beginning where "*Ah receptor*" is the overlapped part.

For the last case: Type E, let us look at another example:

Whereas different anti-CD4 mAb or HIV-1 gp120 could all trigger activation of the protein tyrosine kinases p56lck and p59fyn and phosphorylation of the Shc adaptor protein, which mediates signals to Ras, they differed significantly in their ability to activate NF-AT.

If one system recognized "*protein tyrosine kinases p56lck*" as a protein name and the others recognized "*p56lck*" as a protein name, we called this a Type E and "*p56lck*" the overlapped part.

The integration strategies shown as follows combine the results from two molecular named entity extractors.

- When the protein/gene names produced by two recognizers are completely separate (i.e., type A), we retain each of them as the protein/gene candidates. This integration strategy postulates that one protein (or gene) name recognizer may extract some proteins (or genes) that another recognizer cannot identify.
- When the protein/gene names proposed by two recognizers are exactly the same (i.e., type B), we also retain each of them as the protein/gene candidates. The reason is that when both taggers accept the same protein (or gene) names, there must be some special features that the protein (or gene) names fit.
- When the protein/gene names tagged by two taggers have partial overlap (i.e., types C, D and E), two additional integration strategies are employed, i.e., Yapex-based and KeX-based strategies for proteins, and AB-based and Id-based strategies for genes. In the former strategy, we adopt protein/gene names tagged by Yapex/ABGene as candidates and discard the ones produced by KeX/Idgene. In contrast, the names tagged by KeX/Idgene are kept in the latter strategy. The integration strategy is used because each recognizer has its own characteristics, and we do not know, in advance, which one will perform better. Therefore, we consider one of them as a basis, and then introduce new contributions from another recognizer. That is, if KeX serves as a basis, we choose the tagged names by KeX if any overlaps exist between KeX and Yapex.

5.2 Integration evaluation of proteins

The integration strategies described in Section 5.1 bring together all the possible protein/gene candidates except the ambiguous cases (i.e., types C, D and E). That tends to increase the recall rate. To avoid reducing the precision rate, we also employed the protein/gene

collocates mentioned in Section 3 to filter out the less likely protein/gene candidates. Furthermore, to objectively evaluate the performance of the proposed collocates, we applied our strategies to the test corpus using the terms suggested by human experts on protein evaluation. A total of 48 verbal protein keywords that were used to find the pathway of proteins are listed in Appendix A.

The following four sets of experiments were designed for the Yapex- and KeX-based integration strategies.

(1) YA and KA: The possible protein candidates are merged from the results of the Yapex and KeX systems. If there are any conflicts, the candidates are selected based on either Yapex or KeX. Then, we use the protein collocates automatically extracted in Section 3 to filter out the candidates described in Section 4. That is, we check the co-occurrence of the collocate and protein candidate, no matter which type the protein candidate belongs to.

(2) YB and KB: In the second experiment, we use the terms suggested by human experts for the filtering strategies. YB (KB) is similar to integration strategy YA (KA), except that the collocates are terms suggested by human experts, rather than terms extracted in Section 3.

(3) YA-C and KA-C: If Yapex and KeX recommend the same protein names (i.e., type B), we regard them as protein names, without consideration of the collocates. Otherwise, we use the protein collocates proposed in this study to do the filtering.

(4) YB-C and KB-C: The method is similar to (3) except that the protein collocates are replaced by the terms suggested by human experts.

The experimental results for Yapex-based and KeX-based integration are listed in Tables 5 and 6, respectively. M0 is the baseline model. The named entities proposed by M0 are combined from the results of Yapex and KeX without filtering (i.e., without collocate checking). M0 is used to evaluate the performance changes of the following four cases: without filtering, filtering only, integration only; and both filtering and integration.

The tendencies M32>M31>M2>M1 are still kept in the new experiments. The strategy of delaying the decision until clear evidence found is workable. The performances of YA, YA-C, KA, and KA-C are better than the performances of the corresponding models (i.e., YB, YB-C, KB, and KB-C). This shows that the set of collocates proposed by our system is more complete than the set of terms suggested by human experts. Compared with the recall rate of M0 in Table 3 (i.e., 69.53%), the recall rates of both Yapex- and KeX-based integration are increased, i.e. 77.52% and 70.60%, respectively. This matches our expectations. However, Table 6 shows that the precision rates are reduced more than the increase of the recall rates in

Table 5. Evaluation Results on Yapex-based Integration Strategy

YA	Precision	Recall	F-score	YA-C	Precision	Recall	F-score
M0	61.98%	77.52%	69.75%				
M1	73.56%	71.95%	72.76%	M1	73.83%	74.18%	74.01%
M2	74.98%	72.21%	73.60%	M2	75.93%	75.25%	75.59%
M31	78.84%	75.37%	79.11%	M31	79.42%	76.43%	77.93%
M32	78.81%	76.24%	77.53%	M32	79.40%	76.69%	78.05%
YB				YB-C			
M1	66.79%	44.30%	55.55%	M1	68.92%	58.09%	63.51%
M2	66.79%	44.81%	55.80%	M2	68.78%	58.49%	63.64%
M31	70.20%	65.06%	67.63%	M31	69.07%	69.08%	69.13%
M32	70.19%	65.51%	67.85%	M32	69.07%	69.63%	69.35%

Table 6. Evaluation Results on KeX -based Integration Strategy

KA	Precision	Recall	F-score	KA-C	Precision	Recall	F-score
M0	60.43%	70.60%	65.52%				
M1	66.93%	57.48%	62.21%	M1	67.83%	64.28%	66.06%
M2	66.54%	58.36%	62.45%	M2	67.64%	64.87%	66.26%
M31	67.89%	66.79%	67.34%	M31	66.93%	67.92%	67.43%
M32	67.63%	67.21%	67.42%	M32	66.81%	68.35%	67.58%
KB				KB-C			
M1	67.56%	41.20%	54.38%	M1	69.57%	55.60%	62.59%
M2	66.99%	41.71%	54.35%	M2	69.15%	56.10%	64.06%
M31	69.57%	55.70%	61.64%	M31	68.36%	60.22%	64.29%
M32	69.25%	56.26%	62.76%	M32	68.09%	60.78%	64.44%

some cases. The F-score of KeX-based integration strategy in M1 model is 3.31% worse than that of the baseline M0. This shows that KeX did not perform well in this test set, because it cannot recommend good candidates at the integration stage. Moreover, Table 5 shows that the F-scores of all YA and YA-C models are better than the corresponding models in Table 3 where only the filtering strategies are used. This indicates that Yapex performed better in this test corpus, so that we can enhance the performance by using both the filtering and integration strategies. On the other hand, the F-scores of YB and YB-C are worse than those of M0 in Table 3. This shows that the set of terms suggested by human experts is too weak to improve the performance in the integration strategies. Nevertheless, the models in Tables 6 still cannot compete with M32 in Table 3. The reason may be that some heuristic rules used in Yapex are borrowed from KeX (such as the use of feature terms, e.g., protein, particle and receptor) (Olsson et al., 2002), and added additional filtering strategies (e.g., filtering out names of chemical substances, bibliographical references, chemical formulas, etc.).

5.3 Integration evaluation of genes

We have shown the evaluation results using our integration strategies in the protein domain in Section 5.2. A similar scheme can be applied to the gene domain.

Here, we employ the integration strategies to enlarge the candidate sets, and the gene collocates mentioned in Section 3 to filter out the less likely gene candidates. The terms suggested by human experts are not as complete as the ones our automated method produced. This is demonstrated by the following two sentences.

> The binding capacity and affinity of the *glucocorticoid receptors* were measured and **compared** to clinical data and the plasma cortisol concentrations.
> An over-representation of *T2* in ovarian cancer patients **compared** with controls in the pooled Irish/German population (P<0.025) was observed.

The protein "*glucocorticoid receptors*" and gene "*T2*" are collocated with "compared" which is missed by human experts.

Since the terms suggested by human experts are not as complete as the ones extracted from the corpus, we did not conduct experiments on the terms suggested by human experts in this section. In the following, two sets of experiments for different bases (i.e., ABGene and Idgene), called AB- and Id-based integration strategies, respectively, are conducted.

(1) AB and ID: In these experiments, we use the gene collocates automatically extracted in Section 3 to filter out the candidates merged from the results of ABGene and Idgene.

(2) AB-C and ID-C: If ABGene and Idgene recommend the same gene names, we will select them without consideration of gene collocates. Otherwise, we will use the gene collocates proposed in this study to do the filtering.

The evaluation results of integration strategies on gene domain are listed in Tables 7 and 8.

Table 7. Evaluation Results on AB-based Integration Strategy

AB	Precision	Recall	F-score	AB-C	Precision	Recall	F-score
M0	54.29%	84.47%	69.38%				
M1	64.84%	74.98%	69.91%	M1	67.41%	78.16%	72.78%
M2	65.15%	75.46%	70.31%	M2	67.92%	78.85%	73.39%
M31	67.21%	76.88%	72.05%	M31	69.93%	80.54%	75.24%
M32	68.35%	77.33%	72.84%	M32	69.99%	80.81%	75.40%

Table 8. Evaluation Results on Id-based Integration Strategy

ID	Precision	Recall	F-score	ID-C	Precision	Recall	F-score
M0	31.79%	75.22%	53.51%				
M1	44.62%	66.96%	55.79%	M1	46.31%	68.71%	57.51%
M2	45.03%	67.53%	56.28%	M2	47.29%	69.23%	58.26%
M31	49.16%	68.28%	58.72%	M31	50.44%	70.04%	60.24%
M32	49.74%	69.04%	59.39%	M32	51.71%	70.82%	61.27%

Some results are in agreement with those in the protein experiments. First, the tendencies M32>M31>M2>M1 are still kept in the gene experiments. Second, the recall rates of all models in AB-based integration are increased compared with the recall rate of M0 in Table 4. Third, the results AB-C>AB and ID-C>ID are similar to the results YA-C>YA and KA-C>KA. These results demonstrate that (1) the strategy of delaying the decision until clear evidence is found is useful, (2) the integration strategy is workable for collecting additional correct molecular entities, and (3) if two systems recommend the same biological name, it is an important cue. We now examine Tables 7 and 8 further. Table 7 shows that the precision rates are decreased less than the increase of the recall rates. In contrast, the precision rates are decreased more than the increase of the recall rates shown in Table 8. Idgene-based strategies cannot compete with the M32 strategy in Table 4. This means that the AB-based integration strategy performed well in this test set, but the Id-based integration strategy did not achieve a good performance. In other words, ABGene performed better in this test set than Idgene. Consequently, we infer that ABGene recommended more good candidates than Idgene. The reason may be that ABGene is a general-purpose gene recognizer (Tanabe and Wilbur, 2002) and Idgene focuses on Chinese Gene Variation (Humphreys et al., 2000). Meanwhile, the test set, i.e. the GENIA corpus, covers general documents, rather than documents in some specific topic like Chinese Gene Variation. This leads to the decreased

performance of Idgene, which is worse than ABGene.

6. Annotating Multiple Types of Biomedical Entities

Most of the works in the past on recognizing named entities in the biomedical domain focused on identifying a single type of entities like protein and/or gene names. It is obviously more challenging to annotate multiple types of named entities simultaneously. Intuitively, one can develop a specific recognizer for each type of named entities, run the recognizers one by one to annotate all types of named entities, and merge the results. The problem results from the boundary decision and the annotation conflicts. Instead of constructing five individual recognizers, we regarded the multiple-class annotation as a classification problem, and tried to learn a classifier capable of identifying all the five types of named entities.

Before classification, we have to decide the unit of classification. Since it is difficult to correctly mark the boundary of a name to be identified, the simplest way is to consider an individual word as an instance and assign a type to it. After the type assignment, continuous words of the same type will be marked as a complete named entity of that type. The feature extraction process will be described in the following subsections.

6.1 Feature extraction

The first step in classification is to extract informative and useful features to represent an instance to be classified. In our work, one word is represented by the attributes carried *per se*, the attributes contributed by two surrounding words, and other contextual information. The details are as follows.

6.1.1 Word attributes

The word “attribute” is sometimes used interchangeably with “feature”, but in this report they denote two different concepts. Features are those used to represent a classification instance, and the information enclosed in the features is not necessarily contributed by the word itself. Attributes are defined to be the information that can be derived from the word alone in this paper.

The attributes assigned to each word are whether it is part of a gene/protein name, whether it is part of a species name, whether it is part of a tissue name, whether it is a stop word, whether it is a number, whether it is punctuation, and the part of speech of this word. Instead of using a lexicon for gene/protein name annotation, we employed two gene/protein name taggers, Yapex and GAPSCORE, to do this job. As for part of speech tagging, Brill’s part of speech tagger was adopted.

6.1.2 Context information preparation

Contextual information has been shown helpful in annotating gene/protein names, and therefore two strategies for extracting contextual information at different levels are used. One is the usual practice at a word level, and the other is at a pattern level. Since the training data released in the beginning does not define the abstract boundary, we have to assume sentences are independent of each other and the contextual information extraction was thus limited to be within a sentence.

For contextual information extraction at a word level (Hou and Chen, 2003), collocates along with 4 statistics, including frequency, the average and standard error of distance between word and entity and t-test score, were extracted. The frequency and t-test score were normalized to [0, 1]. Five lists of collocates were obtained for cell-line, cell-type, DNA, RNA, and protein, respectively.

As for contextual information extraction at a pattern level, we first gathered a list of words constituting a specific type of named entities. A hierarchical clustering with cutoff threshold was then performed on the words with edit distance as measure of dissimilarity (see Figure 3). Afterwards, common substrings were obtained to form the list of patterns. With a list of patterns at hand, we estimated the pattern distribution, the occurrence frequencies at and around the current position, given the type of word at the current position. Figure 4 showed an example of the estimated distribution. The average KL-Divergence between any two distributions was computed to discriminate the power of each pattern. The formula is as follows:

$\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j=1, j\neq i}^{n} D(p_i \| p_j)$, where p_i and p_j are the distributions of a pattern given the word at position 0 being *type i* and *j*, respectively.

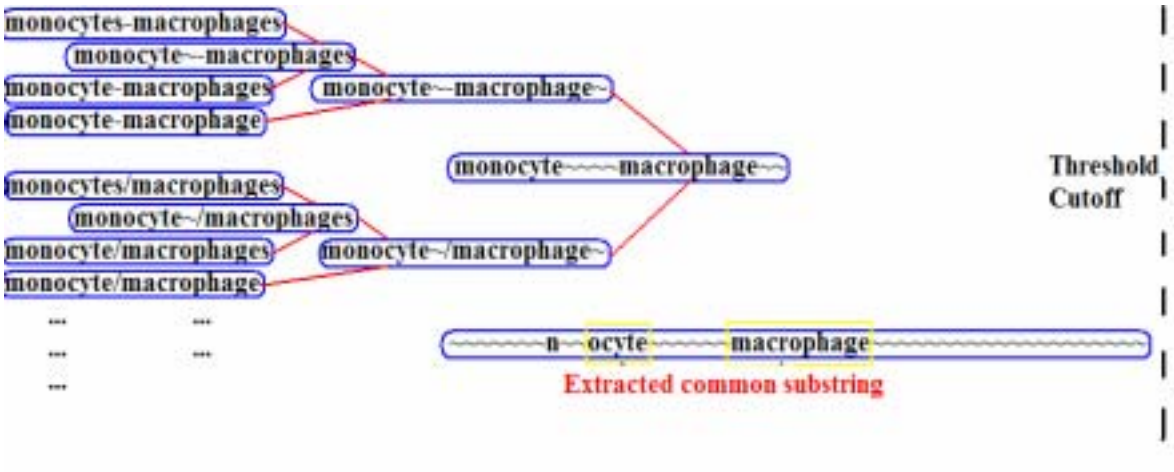

Figure 3. Example of Common Substring Extraction

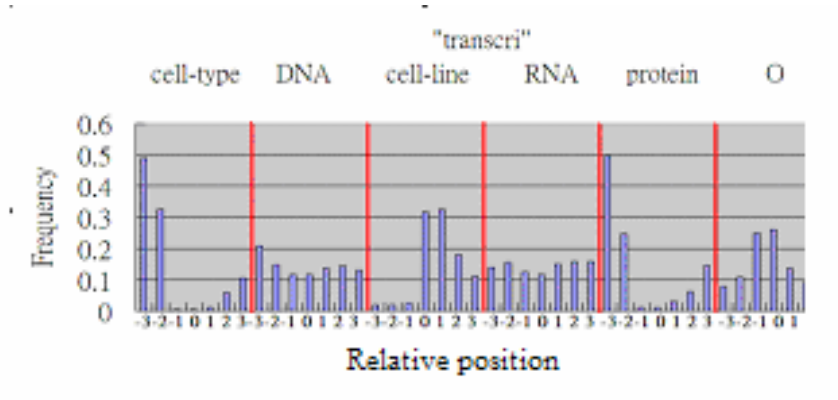

Figure 4. Pattern Distributions Given the Type of Word at Position 0

6.2 Constructing training data

For each word in a sentence, the attributes of the word and the two adjacent words are put into the feature vector. Then, the left five and the right five words are searched for previously extracted collocates. The 15 variables thus added are shown below.

$\sum_{i=-5,i\neq 0}^{5} Freq(w_i \mid type)$

$\sum_{i=-5,i\neq 0}^{5} t-test\_score(w_i \mid type)$

$\sum_{i=-5,i\neq 0}^{5} f(i \mid \hat{\mu}_{w_i,type}, \hat{\sigma}_{w_i,type})$, where f is the pdf for normal distribution, *type* is one of the five types, w_i denotes the surrounding words, $\hat{\mu}_{w_i,type}$ and $\hat{\sigma}_{w_i,type}$ are the maximum likelihood estimates of distance mean and standard deviation for w_i given the type. Next, the left three and right three words along with the current word are searched for patterns, adding 6 variables to the feature vector.

$\sum_{i=-3}^{3} \sum_{p\in P_{w_i}} \text{Prob}_p(i \mid type)$, where *type* is one of the six types including 'O', P_{w_i} is the set of patterns matching w_i, Prob_p denotes the pmf for pattern p. Finally, the type of the previous word is added to the feature vector, mimicking the concept of a stochastic model.

6.3 Classification

Support Vector Machines classification with radial basis kernel was adopted in this task, and the package LIBSVM – A Library for Support Vector Machines (Hsu *et al.*, 2003) was used for training and prediction. The penalty coefficient *C* in optimization and *gamma* in kernel function were tuned using a script provided in this package.

The constructed training data contains 492,551 instances, which is too large for training. Also, the training data is extremely unbalanced (see Table 9) and this is a known problem in SVMs classification. Therefore, we performed stratified sampling to form a smaller and balanced data set for training.

Table 9. Number of Instances for Each Type

Type	# of instances (words)
cell-type	15,466
DNA	25,307
cell-line	11,217
RNA	2,481
protein	55,117
O	382,963

6.4 Results and discussion

Since there is a huge amount of training instances and we do not have enough time to tune the parameters and train a model with all the training instances available, we first randomly selected one tenth and one fourth of the complete training data. The results, as we expected, showed that model trained with more instances performed better (see Table 10). However, we noticed that the performances vary among the 6 types and one of the possible causes is the imbalance of training data among classes (see Table 9). Therefore we decided to balance the training data.

First, the training data was constructed to comprise equal number of instances from each class. However, it didn't perform well and lots of type 'O' words were misclassified, indicating that using only less than 1% of type 'O' training instances is not sufficient to train a good model. Thus two more models were trained to see if the performance can be enhanced. One model has slightly more type 'O' instances than the equally balanced one, and the other model has the ratio among classes being 4:8:4:1:8:16. The results showed increase in recall but drop in precision.

Table 10. Performance of each model (only FULL is shown)

	Model 1/10			**Model 1/4**					
	Recall	Prec.	F-score	Recall	Prec.	F-score	Recall	Prec.	F-score
Full (Object)	0.4756	0.4399	0.4571	0.5080	0.4759	0.4914			
Full (protein)	0.5846	0.4392	0.5016	0.6213	0.4614	0.5296			
Full (cell-line)	0.2420	0.2909	0.2642	0.2820	0.3341	0.3059			
Full (DNA)	0.2784	0.3249	0.2998	0.2888	0.4479	0.3512			
Full (cell-type)	0.3863	0.5752	0.4622	0.4196	0.6115	0.4977			
Full (RNA)	0.0085	0.1000	0.0156	0.0000	0.0000	0.0000			
	Model balanced equally			**Model slightly more 'O'**			**Model 4:8:4:1:8:16**		
Full (Object)	0.1480	0.0990	0.1186	0.1512	0.1002	0.1206	0.5036	0.3936	0.4419
Full (protein)	0.1451	0.1533	0.1491	0.1458	0.1527	0.1492	0.5629	0.4280	0.4863
Full (cell-line)	0.1580	0.0651	0.0922	0.2280	0.0319	0.0560	0.4060	0.2261	0.2904
Full (DNA)	0.1326	0.0466	0.0690	0.1591	0.0582	0.0852	0.3759	0.2457	0.2972
Full (cell-type)	0.1650	0.1375	0.1500	0.1494	0.1908	0.1676	0.4701	0.4900	0.4798
Full (RNA)	0.0932	0.0067	0.0126	0.0169	0.0075	0.0104	0.0593	0.1148	0.0782

After carefully examining the classification results, we found that many of the 'DNA' instances were classified as 'protein' and many of the 'protein' instances were classified as 'DNA'. For example, 904 out of 2,845 'DNA' instances were categorized as 'protein' under 'model 1/4'. The reason may be that Yapex and GAPSCORE do not distinguish gene name from protein names. Even humans don't do very well at this (Krauthammer *et al.*, 2002).

We originally planned to verify if the tag of the previous word is useful and to obtain the results assuming the previous word is always correctly predicted. Because the previous word tag is predicted with our classifier, this introduced a lot of noise.

7. Extracting Gene References into Function

7.1 Architecture overview

A complete annotation system may be done at two stages, including (1) extraction of molecular function for a gene from a publication and (2) alignment of this function with a GO term. Figure 5 shows an example. The left part is an MEDLINE abstract with the function description highlighted. The middle part is the corresponding Gene References into Function (GeneRIF). The matching words are in bold, and the similar words are underlined. The right part is the GO annotation. This figure shows a possible solution of maintaining the knowledge bases and ontology using natural language processing technology. We addressed automation of the first stage in this report.

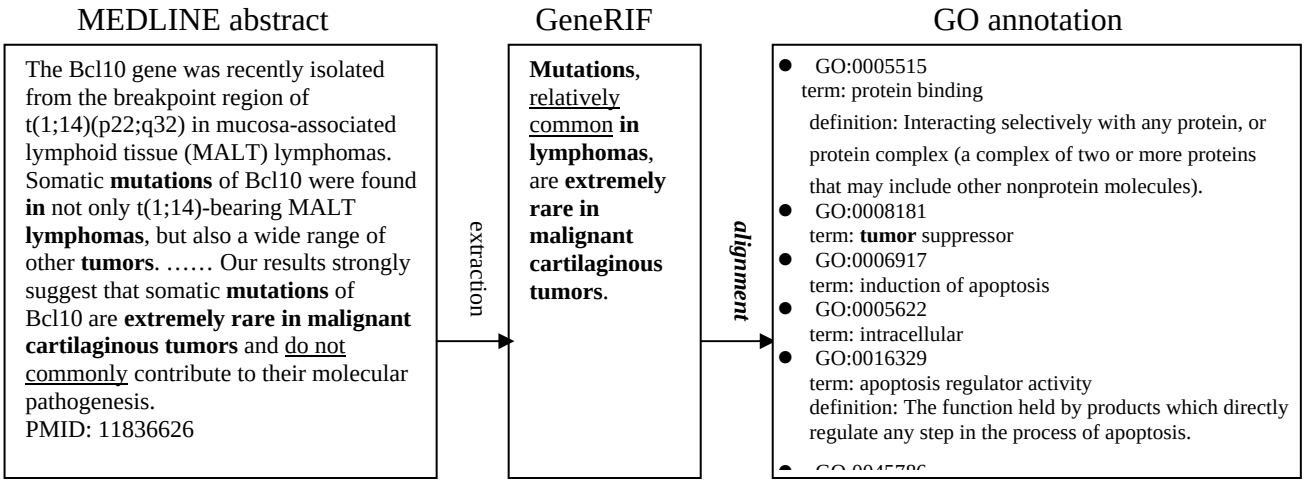

Figure 5. An Example of Complete Annotation from a Literature to Gene Ontology

The overall architecture is shown in Figure 6. First, we constructed a training corpus in such a way that GeneRIFs were collected from LocusLink and the corresponding abstracts were retrieved from MEDLINE. "GRIF words" and their weights were derived from the training corpus. Then Support Vector Machines were trained using the derived corpus. Given a new abstract, a sentence is selected from the abstract to be the candidate GeneRIF.

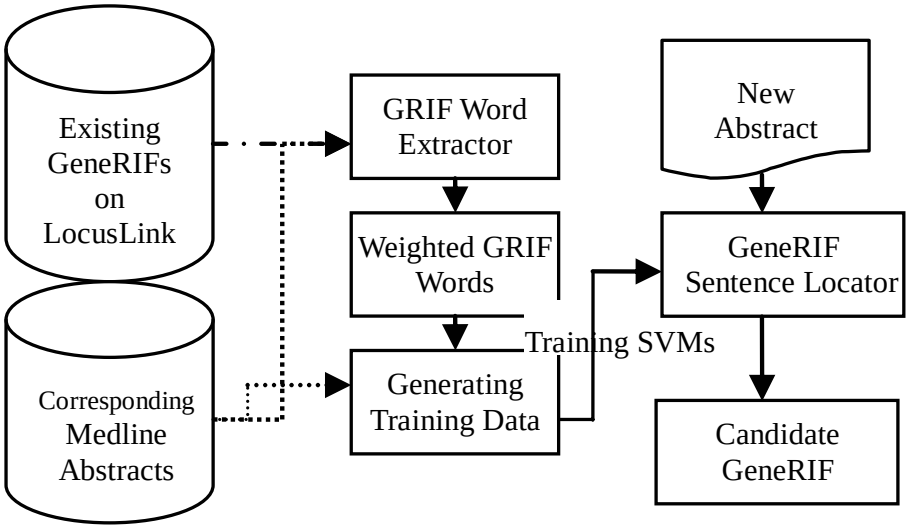

Figure 6. Architecture of Extracting Candidate GeneRIF

7.2 Methods

We adopted several weighting schemes to locate the GeneRIF sentence in an abstract in the official runs (Hou *et al*., 2003). Inspired by the work by Jelier *et al*. (2003), we incorporated their definition of classes into our weighting schemes, converting this task into a classification problem using SVMs as the classifier. We ran SVMs on both sets of features proposed by Hou *et al*. (2003) and Jelier *et al*. (2003), respectively. Finally, all the features were combined and some feature selection methods were applied to train the classifier.

7.2.1 Training and test material preparation

Since GeneRIFs are often cited verbatim from abstracts, we decided to reproduce the GeneRIF by selecting one sentence in the abstract. Therefore, for each abstract in our training corpus, the sentence most similar to the GeneRIF was labelled as the GeneRIF sentence using Classic Dice coefficient as similarity measure. Totally, 259,244 abstracts were used, excluding the abstracts for testing. The test data for evaluation are the 139 abstracts used in TREC 2003 Genomics track.

7.2.2 GRIF words extraction and weighting scheme

We called the matched words between GeneRIF and the selected sentence as *GRIF words* in this paper. GRIF words represent the favorite vocabulary that human experts use to describe gene functions. After stop words removal and stemming operation, 10,506 GRIF words were extracted.

In our previous work (Hou *et al.*, 2003), we first generated the weight for each GRIF word. Given an abstract, the score of each sentence is the sum of weights of all the GRIF words in this sentence. Finally, the sentence with the highest score is selected as the candidate GeneRIF. This method is denoted as OUR weighting scheme, and several heuristic weighting schemes were investigated. Here, we only present the weighting scheme used in classification. The weighting scheme is as follows. For GRIF word *i*, the number of occurrence n_i^G in all the GeneRIF sentences and the number of occurrence n_i^A in all the abstracts were computed and n_i^G / n_i^A was assigned to word *i* as its weight.

7.2.3 Class definition and feature extraction

The distribution of GeneRIF sentences showed that the position of a sentence in an abstract is an important clue to where the answer sentence is. Jelier *et al.* (2003) considered only the title, the first three and the last five sentences, achieving the best performance in TREC official runs. Their Naïve Bayes model is as follows. An abstract *a* is assigned a class v_j by calculating v_{NB}:

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \times \prod_{i \in S} \prod_{k \in W_{a,i}} P(w_{k,i} \mid v_j)$$

where v_j is one of the nine positions aforementioned, *S* is the set of 9 sentence positions, $W_{a,i}$ is the set of all word positions in sentence *i* in abstract *a*, $w_{k,i}$ is the occurrence of the normalized word *k* in sentence *i* and *V* is the set of 9 classes.

We, therefore, represented each abstract by a feature vector composed of the scores of 9 sentences. Furthermore, with a list of our 10,506 GRIF words at hand, we also computed the occurrences of these words in each sentence, given an abstract. Each abstract is then represented by the number of occurrences of these words in the 9 sentences respectively, i.e., the feature vector is 94,554 in length. Classification based on this type of features is denoted the *sentence-wise bag of words model* in the rest of this paper. Combining these two models, we got totally 94,563 features.

Since we are extracting sentences discussing gene functions, it's reasonable to expect gene or protein names that appeared in the GeneRIF sentence. Therefore, we employed Yapex (Olsson *et al.*, 2002) and GAPSCORE (Chang *et al.*, 2004) protein/gene name detectors to count the number of protein/gene names in each of the 9 sentences, resulting in 94,581 features.

7.2.4 Training SVMs

The whole process related to SVM is done via LIBSVM – A Library for Support Vector Machines (Hsu *et al.*, 2003). Radial basis kernel was adopted based on our previous experience. However, further verification showed that the combined model with either linear or polynomial kernel only slightly surpassed the baseline, attaining 50.67% for CD. In order to get the best-performance classifier, we tuned two parameters, C and gamma. They are the penalty coefficient in optimization and a parameter for the radial basis kernel, respectively. Four-fold cross validation accuracy was used to select the best parameter pair.

7.2.5 Picking up the answer sentence

Test instances are first fed to the classifier to get the predicted positions of GeneRIF sentences. In case that the predicted position doesn't have a sentence, which happens when the abstract doesn't have enough sentences, the sentence with the highest score is picked for the weighting scheme and the combined model, and the title is picked for the sentence-wise bag of words model.

7.3 Results and Discussion

The performance measures are based on Dice coefficient, which calculates the overlap between the candidate GeneRIF and actual GeneRIF. Classic Dice (CD) is the classic Dice formula using a common stop word list and the Porter stemming algorithm. Due to lack of space, we referred you to the Genomics track overview for the other three modifications of CD (Hersh and Bhupatiraju, 2003).

The evaluation results are shown in Table 11.

Table 11. Comparison of performances on the 139 abstracts

		CD	MUD	MBD	MBDP
1	Jelier (Sentence-wise bag of words + Naïve Bayes)	57.83%	59.63%	46.75%	49.11%
2	Sentence-wise bag of words + SVMs	58.92%	61.46%	47.86%	50.84%
3	OUR Weighting scheme	50.18%	46.71%	33.47%	38.83%
4	OUR Weighting scheme + SVMs	56.86%	58.81%	45.08%	48.10%
5	**Combined**	**59.51%**	**62.16%**	**48.17%**	**51.25%**
6	Combined + gene/protein names	57.59%	59.95%	46.69%	49.68%
7	Combined + BWRatio feature selection	57.59%	59.90%	47.11%	50.08%
8	Combined + Graphical feature selection	58.81%	61.09%	47.98%	50.92%
9	Optimal Classifier	67.60%	70.74%	59.28%	62.09%
10	Baseline	50.47%	52.60%	34.82%	37.91%

The first row shows the official run of Jelier's team, the first place in the official runs. The second row shows the performance when the Naïve Bayes classifier adopted by Jelier is replaced with SVMs. The third row is the performance of our weighting scheme without a classifier. The fourth row then lists the performance when our weighting scheme is combined with SVMs. The fifth row is the result when our weighting scheme and the

sentence-wise bag of words model are combined together. The sixth row is the result when two gene/protein name detectors are incorporated into the combined model. The next two rows were obtained after two feature selection methods were applied. The ninth row shows the performance when the classifier always proposes a sentence most similar to the actual GeneRIF. The last row lists the baseline, i.e., a title is always picked.

A comparative study on text categorization (Joachims, 1998) showed that SVMs outperform other classification methods, such as Naïve Bayes, C4.5, and k-NN. The reasons would be that SVMs are capable of handling large feature space, text categorization has few irrelevant features, and document vectors are sparse. The comparison between SVMs and the Naïve Bayes classifier again demonstrated the superiority of SVMs in text categorization (rows 1, 2).

The performance greatly improved after introducing position information (rows 3, 4), showing the sentence position plays an important role in locating the GeneRIF sentence. The 2% difference between rows 2 and 4 indicates that the features under sentence-wise bag of words model are more informative than those under our weighting scheme. However, with only 9 features, our weighting scheme with SVMs performed fairly well. Comparing the performance before and after combining our weighting scheme and the sentence-wise bag of words model (rows 2, 5 and rows 4, 5), we can infer from the performance differences that both models provide mutually exclusive information in the combined model. The result shown in row 6 indicates that the information of gene/protein name occurrences did not help identify the GeneRIF sentences in these 139 test abstracts.

We performed feature selection on the combined model to reduce the dimension of feature space. There were two methods applied: a supervised heuristic method (denoted as BWRatio feature selection in Table 2) (S. Dutoit *et al.*, 2002) and another unsupervised method (denoted as Graphical feature selection in Table 2) (Chang *et al.*, 2002). The number of features was then reduced to about 4,000 for both methods. Unfortunately, the performance did not improve after either method was applied. This may be attributed to over-fitting training data, because the cross-validation accuracies are indeed higher than those without feature selection. The result may also imply there are little irrelevant features in this case.

8. Concluding Remarks

Table 12 summarizes the results of enhancing the performance of protein and gene name recognizers with filtering and integration strategies. We propose a fully automatic method of mining collocates from scientific texts in the protein and gene domains, and employ the extracted collocates to improve the precision rate of protein/gene name recognition. The precision of Yapex is increased from 70.90% to 85.84% at a small expense in the recall rate (i.e. it only decreases 2.44%) when collocates are incorporated. When the integration-only approach is adopted (i.e. -filtering, +integration), the F-score of the Yapex-based (ABGene-based) integration is a little lower than that of the filtering-only approach (i.e. +filtering, -integration). This shows that collocation learning is useful, and integration depends on the individual performance NE recognizers. When both filtering and integration (i.e. +filtering, +integration) strategies are employed together, the Yapex-based integration with KeX achieves 7.83% F-score increase compared to the pure Yapex method (i.e., -filtering, -integration). The ABGene-based integration with Idgene shows a 10.18% F-score increase relative to the pure ABGene method.

Table 12. Summary of Experimental Results for enhancing performance of protein and gene name recognizers

Strategy / Performance of protein/gene		-filtering -integration	+filtering -integration	-filtering +integration	+filtering +integration
Protein	Precision	70.90%	85.84%	61.98%	79.40%
	Recall	69.53%	67.09%	77.52%	76.69%
	F-Score	70.22%	76.47%	69.75%	78.05%
Gene	Precision	55.87%	70.08%	54.29%	69.99%
	Recall	74.56%	71.89%	84.47%	80.81%
	F-Score	65.22%	70.99%	69.38%	75.40%

The main benefits of our method are: (1) The collocates used in the filtering strategies are produced by the training corpus rather than by intuition. This forms a more complete set than one identified by human experts; (2) The combination of the filtering and integration strategies shows better performance than the original protein/gene name taggers. The main drawback of our method is that we cannot solve the problem of false negatives. To solve such problems, more linguistic technologies need to be investigated in order to recover the false negatives. In addition, the performance of integrity strategies relied on the performance of the selected taggers as shown in Table 12.

This tendency is consistent with gene and protein name entity extraction. We expect that the methodologies can be easily extended to other domains, such as drugs and diseases. This will be verified in future work. The protein (or gene) collocates extracted from the domain corpus are also important keywords for pathway discovery, so that a systematic way from basic named entities finding to the discovery of complex relationships can be explored. Although the relation extraction involves more complex issues, such as related objects, pathway direction and dependency relation, the correct recognition of genome/protein is the most basic task and this can be help with our methods. The values of the frequency, average distance, standard deviation and t-score can serve as some features for machine learning approaches to tag the protein/gene names. This will be studied. The experimental systems adopted in this paper are rule-based. The effects of combining different types of protein/gene name taggers, e.g., rule-based and corpus-based, will be investigated in the future.

In the second study of annotating multiple types of biological entities, we introduced the use of existing taggers and presented a way to collect common substrings shared by entities. Due to lack of time, the models were not well tuned against the two parameters – C and gamma, influencing the capabilities of the models. Further, not all of the training instances provided were used to train the model, and it will be interesting and worthwhile to investigate. How to deal with data imbalance is another important issue. By solving this problem, further evaluation of feature effectiveness would be facilitated. We believe there is much left for our approach to improve and it may perform better if more time is given.

For the last application of extracting GeneRIF from biological documents, we proposed an automatic approach to locate the GeneRIF sentence in an abstract with the assistance of SVMs, reducing the human effort in updating and maintaining the GeneRIF field in the LocusLink database.

We have to admit that the 139 abstracts provided in TREC 2003 are too few to verify the performance among models, and the results based on these 139 abstracts may be slightly biased. Our next step would aim at measuring the cross-validation performances using Dice coefficient.

The syntactic information is worth exploring, since the sentences describing gene functions may share some common structural patterns. Moreover, how the weighting scheme affects the performance is also very interesting. We are currently trying to obtain a weighting scheme that can best distinguish GeneRIF sentence from non-GeneRIF sentence without classifiers.

Acknowledgements

Part of this research was supported in part by National Science Council under contracts NSC-91-2213-E-002-088, and NSC-92-2213-E-002-022. We wish to thank Dr. Lorrie Tanabe and Dr. W. John Wilbur in NCBI, NLM, NIH, and Dr. George Demetriou in the Department of the Computer Science, University of Sheffield who kindly supported the resources in this work.

References

Adamic L.A., Wilkinson D., Huberman B.A. and Adar E. (2002) A Literature Based Method for Identifying Gene-Disease Connections. IEEE Computer Society Bioinformatics Conference (CSB'02) 2002; 109-117.

BIOSIS organization (1999). Biomedical Literature Searching: A Comparison of BIOSIS Previews, EMBASE, and MEDLINE. BIOSIS Evolutions 1999; 6(3): 1, 4-7.

Blaschke C., Andrade M.A., Ouzounis C. and Valencia A. (1999) Automatic Extraction of Biological Information from Scientific Text: Protein-Protein Interactions. Proceedings of 7th International Conference on Intelligent Systems for Molecular Biology 1999; 60-67.

Bhalotia G., Nakov P.I., Schwartz A.S., and Hearst M.A. (2003) BioText Team Report for the TREC 2003 Genomics Track. TREC 2003 work notes 2003; 158-166.

Brill E. (1994) Some Advances in Transformation-Based Part of Speech Tagging. Proceedings of the National Conference on Artificial Intelligence. AAAI Press; 1994, p. 722-727.

Burges C. (1998) A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2: 121-167.Chen H.H.; Ding Y.W. and Tsai S.C. Named Entity Extraction for Information Retrieval. Computer Processing of Oriental Languages. Special Issue on Information Retrieval on Oriental Languages 1998; 12(1): 75-85.

Chang J.T., Schutze H. and Altman R.B. (2004) GAPSCORE: Finding Gene and Protein Names One Word at a Time. Bioinformatics 2004; 20(2): 216-225.

Chang Y.C., Hsu I.H.and Chou. L.Y. (2002) Graphical Features Selection Method. Intelligent Data Engineering and Automated Learning, Edited by H. Yin, N. Allinson, R. Freeman, J. Keane, and S. Hubband, 2002.

Chen H.H. and Lee J.C. (1996) Identification and Classification of Proper Nouns in Chinese Texts. Proceedings of 16th International Conference on Computational Linguistics 1996; 222-229.

Collier N., Nobata C. and Tsujii J.I. (2000) Extracting the Names of Genes and Gene Products with a Hidden Markov Model. Proceedings of 18th International Conference on Computational Linguistics 2000; 201-207.

Collier N., Park H.S., Ogata N., Tateishi Y., Nobata C. and Ohta T. (1999) The GENIA project: Corpus-based Knowledge Acquisition and Information Extraction from Genome Research Papers. Proceedings of the Annual Meeting of the European Chapter of the Association for Computational Linguistics (EACL'99) 1999, June.

Craven M. and Kumlien J. (1999) Constructing Biological Knowledge Bases by Extracting Information from Text Sources. Proceedings of 7th International Conference on Intelligent Systems for Molecular Biology 1999; 77-86.

DARPA (1998) Proceedings of 7th Message Understanding Conference 1998.

Dutoit S., Yang Y.H., Callow M.J. and Speed T.P. (2002) Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. J. Amer. Statis. Assoc. 2002; 97:77-86.

Fan J.W. (2003) Information Retrieval and Extraction for the Chinese Gene Variation Database (CGVdb). Unpublished Master Thesis; 2003.

Fox C. (1992) Lexical Analysis and Stoplists. In: Frakes W. B. and Baeza-Yates R. editors. Information Retrieval: Data Structures and Algorithms. Prentice Hall; 1992; 102-130.

Friedman C., Kra P., Yu H., Krauthammer M. and Rzhetsky A. (2001) GENIES: A Natural Language Processing System for the Extraction of Molecular Pathways from Journal Articles. Bioinformatics 2001; 17(S1): 74-82.

Fukuda K., Tsunoda T., Tamura A. and Takagi T. (1998) Toward Information Extraction: Identifying Protein Names from Biological Papers. Proceedings of Pacific Symposium on Biocomputing 1998; 707-718.

GENIA project. http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA.

Hanisch D, Fluck J, Mevissen H.T. and Zimmer R. (2003) Playing Biology's Name Game: Identifying Protein Names in Scientific Text. Proceedings of the Pacific Symposium on Biocomputing 2003; 403-414.

Hersh W. and Bhupatiraju R.T. (2003) TREC Genomics Track Overview. Proceedings of TREC 2003.

Hirschman L., Park J.C., Tsujii J., Wong L. and Wu C.H. (2002) Accomplishments and Challenges in Literature Data Mining for Biology. Bioinformatics 2002; 18(12): 1553-1561.

Hou W.J. and Chen H.H. (2002) Extracting Biological Keywords from Scientific Text. Proceedings of 13th International Conference on Genome Informatics 2002; 571-573.

Hou W.J. and Chen H.H. (2003) Enhancing Performance of Protein Name Recognizers Using Collocation. Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine 2003; 25-32.

Hou W.J., Teng C.Y., Lee C. and. Chen H.H. (2003) SVM Approach to GeneRIF Annotation. Proceedings of TREC 2003.

Hsu C.W., Chang C.C and Lin C.J. (2003) A Practical Guide to Support Vector Classification. http://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html.

Humphreys K., Demetriou G. and Gaizauskas R. (2000) Two Applications of Information Extraction to Biological Science Journal Articles: Enzyme Interactions and Protein Structures. Proceedings of Pacific Symposium on Biocomputing 2000; 5: 502-513.

Jelier R., Schuemie M., Eijk C.V.E., Weeber M., Mulligen E.V., Schijvenaars B., Mons B. and Kors J. (2003) Searching for geneRIFs: concept-based query expansion and Bayes classification. Proceedings of TREC 2003; 167-174.

Joachims T. (1998) Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Proceedings of ECML-98 1998; 137-142.

Kazama J., Makino T., Ohta Y. and Tsujii J. (2002) Tuning Support Vector Machines for Biomedical Named Entity Recognition. Proceedings of the ACL 2002 workshop on NLP in the Biomedical Domain 2002; 1-8.

Krauthammer M, Rzhetsky A, Morozov P and Friedman C. (2000) Using BLAST for Identifying Gene and Protein Names in Journal Articles. Gene 2000; 259(1-2): 245-252.

Lee K.J., Hwang Y.S. and Rim H.C. (2003) Two-Phase Biomedical NE Recognition based on SVMs. Proceedings of the ACL 2003 Workshop on NLP in Biomedicine 2003; 33-40.

Manning C.D. and Schutze H.(1999) Foundations of Statistical Natural Language Processing. The MIT Press; 1999.

Marcotte E.M., Xenarios I. and Eisenberd D. Mining Literature for Protein-protein Interactions. Bioinformatics 2001; 17(4): 359-363.

Morgan A., Hirschman L., Yeh A. and Colosimo M. Gene Name Extraction Using FlyBase Resources. Proceedings of the ACL 2003 Workshop(1999) on Natural Language Processing in Biomedicine 2003; 1-8.

Ng S.K. and Wong M. (1999) Toward Routine Automatic Pathway Discovery from On-line Scientific Text Abstracts. Proceedings of 10th International Conference on Genome Informatics 1999; 104-112.

Olsson F., Eriksson G., Franzen K., Asker L. and Liden P. (2002) Notions of Correctness when Evaluating Protein Name Taggers. Proceedings of the 19th International Conference on Computational Linguistics 2002; 765-771.

Ono T., Hishigaki H., Tanigami A. and Takagi T. (2001) Automated Extraction of Information on Protein-Protein Interactions from the Biological Literature. Bioinformatics 2001; 17(2): 155-161.

Park J.C., Kim H.S. and Kim J.J. (2001) Bidirectional Incremental Parsing for Automatic Pathway Identification with Combinatory Categorial Grammar. Proceedings of Pacific Symposium on Biocomputing 2001; 6: 396-407.

Pearson H. (2001) Biology's Name Game. Nature 2001; 411: 631-632.

Pruitt K.D., Katz K.S., Sicotte H. and Maglott D.R. (2000) Introducing RefSeq and LocusLink: Curated Human Genome Resources at the NCBI. Trends Genet 2000; 16(1): 44-47.

Ratnaparkrhi. A. (1998) Maximum Entropy Models for Natural Language Ambiguity Resolution. PhD Thesis, University of Pennsylvania; 2003.

Rindflesch T.C., Tanabe L., Weinstein J.N. and Hunter L. (2000) EDGAR: Extraction of Drugs, Genes, and Relations from Biomedical Literature. Proceedings of Pacific Symposium on Biocomputing 2000; 5: 517-528.

Sekimizu T., Park H.S. and Tsujii T. (1998) Identifying the Interaction Between Genes and Gene Products Based on Frequently Seen Verbs in Medline Abstracts. Genome Informatics 1998; 62-71.

Takeuchi K. and Collier N. (2003) Bio-Medical Entity Extraction using Support Vector Machines. Proceedings of the ACL 2003 workshop on NLP in Biomedicine 2003; 57-64.

Tanabe L. and Wilbur W.J. (2002) Tagging Gene and Protein Names in Biomedical Text. Bioimformatics 2002; 18(8): 1124-1132.

Thomas J., Milward D., Ouzounis C., Pulman S. and Carroll M. (2000) Automatic Extraction of Protein Interactions from Scientific Abstracts. Proceedings of Pacific Symposium on Biocomputing 2000; 5: 538-549.

TREC 2003 Genome TRACK, http://medir.ohsu.edu/~genomics/.

Tsuruoka Y. and Tsujii J. (2003) Boosting Precision and Recall of Dictionary-based Protein Name Recognition. Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine 2003; 41-48.

Wong L. (2001) PIES, a Protein Interaction Extraction System. Proceedings of Pacific Symposium on Biocomputing 2001; 6: 520-531.

Yamamoto K., Kudo T., Konagaya A. and Matsumoto Y. (2003) Protein Name Tagging for Biomedical Annotation in Text. Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine 2003; 65-72.

Appendix A: Terms suggested by an expert

accompan (-ied, -ies, -y, -ying),
activat (-e, -ed, -es, -ing, -ion, -or, -ors, -ory),
affect (-, -ed, -ing, -s),
aggregat (-e, -ed, -es, -ing, -ion),
assembl (-e, -ed, -es, -ing, -y),
associat (-e, -ed, -es, -ing, -ion),
attract (-, -ed, -ing, -ion, -s),
bind (-, -ing, -s) / bound,
catalys (-e, -ed, -es, -ing, -tic),
catalyz (-e, -ed, -es, -ing),
cluster (-, -ed, -ing, -s),
communicat (-e, -ed, -es, -ing, -ion),
complex (-, -ed, -es, -ing),
construct (-, -ed, -ing, -ion, -s),
control (-, -ed, -ing, -led, -ling, -s),
cooperat (-e, -ed, -es, -ing, -ion, -or, -ors),
correlat (-e, -ed, -es, -ing, -ion),
coupl (-e, -ed, -es, -ing),
crosslink (-, -ed, -ing, -s),
deglycosylat (-e, -ed, -es, -ing, -ion, -ory),
demethylat (-e, -ed, -es, -ing, -ion, -ory),
dephosphorylat (-e, -ed, -es, -ing, -ion, -ory),
effect (-, -ed, -ing, -s),
eliminat (-e, -ed, -es, -ing, -ion),
enabl (-e, -ed, -es, -ing),
enhanc (-e, -ed, -er, -es, -ing),
glycosylat (-e, -ed, -es, -ing, -ion, -ory),
group (-, -ed, -ing, -s),
help (-, -ed, -ing, -s),
hinder (-, -ed, -ing, -s),
inactivat (-e, -ed, -es, -ing, -ion, -or, -ors, -ory),
inhibit (-, -ed, -ing, -ion, -or, -ors, -ory, -s),
integrat (-e, -ed, -es, -ing, -ion),
interact (-, -ed, -ing, -ion, -s),
link (-, -ed, -ing, -s),
methylat (-e, -ed, -es, -ing, -ion),
obstacl (-e, -ed, -es, -ing),
participat (-e, -ed, -es, -ing, -ion),
phosphorylat (-e, -ed, -es, -ing, -ion, -ory),
prim (-e, -ed, -es, -ing),
process (-, -ed, -es, -ing),
react (-, -ed, -ing, -ion, -or, -ors, -ory, s),
regulat (-e, -ed, -es, -ing, -ion, ,-or, -ory),
relat (-e, -ed, -es, -ing, -ion),
signal (-, -ed, -ing, , -led, -ling, -s),
stimulat (-e, -ed, -es, -ing, -ion, ,-or, -ory),
suppress (-, -ed, -es, -ing, -ion),

transduc (-e, -ed, -es, -ing, -tion, ,-tor, -tory),
trigger (-, -ed, -ing, -s)