BIOLOGICAL INFORMATION EXTRACTION USING PATTERNS OF CHARACTERS, TAG SEQUENCES AND SUBGRAPHS

by

Haibin Liu

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
November 2010

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "BIOLOGICAL INFORMATION EXTRACTION USING PATTERNS OF CHARACTERS, TAG SEQUENCES AND SUBGRAPHS" by Haibin Liu in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dated: November 8, 2010

External Examiner: Dr. Lawrence Hunter

Research Supervisors: Dr. Vlado Kešelj

Dr. Christian Blouin

Examining Committee: Dr. Evangelos E. Milios

Dr. Robert Beiko

Departmental Representative:

DALHOUSIE UNIVERSITY

DATE: November 8, 2010

AUTHOR: Haibin Liu

TITLE: BIOLOGICAL INFORMATION EXTRACTION USING PATTERNS OF CHARACTERS, TAG SEQUENCES AND SUBGRAPHS

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: Ph.D. CONVOCATION: May YEAR: 2011

Signature of Author

Table of Contents

List of Tables

List of Figures

Abstract

The magnitude of the document collection in the biology domain boosts the demand for effective and efficient literature mining and knowledge discovery that can help biologists to gather and make use of the knowledge encoded in text documents. In this thesis, we present three different pattern-based techniques to target two important tasks of biological information extraction: entity recognition and relation extraction.

The first technique is an unsupervised method to automatically extract domain-specific prefix and suffix characters from biological corpora. The extracted characters are integrated into the parametrization of an existing system for biological entity recognition in order to aid the system to annotate biological entities.

The second technique is an approach to identify sentences that describe interactions between co-occurring biological entities using patterns defined as a sequence of specialized Part-of-Speech (POS) tags that capture the structure of key sentences in the scientific literature. Each candidate sentence for the classification task is encoded as a POS array and then aligned to a collection of pre-extracted patterns. The quality of the alignment is expressed as a pairwise alignment score. The most innovative component of this work is the use of a Genetic Algorithm (GA) to maximize the classification performance of the alignment scoring scheme.

The third technique is a graph matching-based approach to extract complex biological events from the scientific literature. Sentences are represented as dependency graphs, and biological event rules are extracted from sentences as minimal dependency graphs that capture the typical contextual structures of biological events. We investigate whether the subgraph matching problem can be used in the BioNLP field to extract biological events by searching for subgraphs isomorphic to the graphs of event rules within the graphs of sentences.

Acknowledgements

This work has been five years in the making. I would like to acknowledge my two supervisors, Dr. Vlado Kešelj and Dr. Christian Blouin. They were both excellent mentors and friends to me. Although busy with their own work, they went out of their way to provide exceptional guidance, creative ideas and suggestions. They generously accommodated my need to spend time with my family in Denver, such as patiently holding teleconferences with me. I benefited deeply from Dr. Kešelj's broad knowledge of computer science, and strong math background. Dr. Blouin provided profound thoughts and insights in molecular biology. I could not have done this work without their support.

I would also like to thank my wife Jinyan. She has been tremendously supportive, loving, and patient. I understand that life was not always easy for her when I was absent, working in Halifax. I know as long as she is by my side, I can accomplish anything. I would also like to thank her for bringing our loving and joyful daughter into this world. Our one-year-old daughter Arwen was also involved with my work and when she is older I will tell her about her contribution. One day as I worked, surrounded by papers, she entered the room, grabbed a page and ran away. She mischievously waved the paper in the air as I chased her. Finally, I would like to thank my parents for their unwavering faith in me, and their emotional and financial support. I felt I could always turn to them. They were an incredible source of strength to me throughout this process.

This work was made possible in part by funding from NSERC and Faculty of Computer Science at Dalhousie University. Their contributions made it easier for me to concentrate on my work.

Chapter 1

Introduction

In this thesis, we explore three different pattern-based techniques to extract biological entities, relations and events in the molecular biology literature. Firstly, we propose an unsupervised method to automatically extract domain-specific prefix and suffix characters from biological corpora. The extracted characters are integrated into the parametrization of an existing system for biological entity recognition in order to aid the system in annotating biological entities. Secondly, we propose an approach to identify sentences that describe interactions between co-occurring biological entities from patterns. A pattern is defined as a sequence of specialized Part-of-Speech (POS) tags that capture the structure of key sentences in the scientific literature. Thirdly, we propose a graph matching-based approach to extract biological events from biological texts with the same objectives as these of the BioNLP'09 shared task [1]. In this section, sentences and rules of biological event are represented as dependency graphs. The event extraction process corresponds to matching the graphs of event rules within the graphs of sentences.

The chapter is organized as follows: In section 1.1, we present the motivation of the thesis. Our objectives are stated in section 1.2. The outline of the thesis is given in section 1.3.

1.1 Motivation

Molecular biology is at the forefront of the life science and is of unprecedented complexity. The amount of new discoveries as published in the scientific literature in the biological domain is growing at an unmanageable pace. PubMed is a service of the National Library of Medicine (NLM) that provides access to MEDLINE [2], the bibliographic database of National Library of Medicine that contains indexed citations, abstracts and full text articles on life sciences and biomedical topics. PubMed now has records from more than 4,800 journals accounting for approximately 19.8 million

citations [2], and is increasing its size at the speed of approximately two papers per minute [3]. This growth makes it practically impossible for molecular biologists to comprehensively cover all of the literature related to their field. Most biological facts are available in the unstructured text of scientific articles [4–6]. Therefore, it becomes very difficult to extract the core information in an efficient manner.

The magnitude of the document collection in the molecular biology domain boosts the demand for effective and efficient literature mining and knowledge discovery that can help molecular biologists to gather and make use of the knowledge from text documents. In order to make organized and structured information available, reliable and robust biological information extraction becomes critical. Beginning with natural language text from a biological document source, the goal is to identify biological entities and relations of specific predefined classes in the text, and represent the extracted information in an organized manner [7]. Biological information extraction should enhance researchers' ability to extract information from the growing corpus of the biological literature by making the process of analyzing texts more efficient and comprehensive. Since the information acquired from the literature becomes more structured through the automated processing, it can be integrated with other information. Biological information extraction involves two important tasks that are closely related to each other: biological entity recognition and biological relation extraction.

Biological entity recognition is a preparatory step in information extraction for the biological sciences. Proteins, genes, and diseases are examples of relevant entities in this task. Considering the structure, there are two types of entities: single word entities and multi-word entities. Recognizing biological entities from texts allows to capture their underlying meaning and further extract semantic relationships between entities and other useful information. Many systems [6, 8–15] have been proposed to annotate biological entities based on different methodologies in which determining entity boundaries is usually the first task. It has been demonstrated, however, that accurately locating entity boundaries is difficult [16]. This is so because of the ambiguity of entities, and the peculiarity of the language used in the biological literature.

Systems based on machine learning techniques use training data to learn features useful for differentiating biological entities from non-biological entities. Domain-specific prefix and suffix characters are reported as one of the most effective features in aiding the systems to annotate biological entities. Although various approaches have been employed to acquire biological prefixes and suffixes, they all have different limitations [14, 16].

In this thesis, we propose an unsupervised method to automatically extract domain-specific prefix and suffix characters from biological corpora, and explore their use in the task of biological entity recognition.

Biological relation extraction focuses on the automated extraction of semantic relations between the recognized biological entities from the scientific literature. Automatic relation extraction has a broad range of applications in molecular biology, including support for the creation and annotation of pathways, automatic population or enrichment of databases, and the formulation of new hypothesis. Relation extraction systems can be trained to recognize a wide range of activities, such as protein-protein interactions, subcellular localizations, gene regulatory events, and metabolic or signaling reactions.

One of the most straightforward relation mining approaches is the co-occurrence search. The assumption is that for describing an interaction between two entities their names usually occur in the same text or part of the text. However, co-occurrence certainly does not guarantee that a passage contains an interaction [5, 17, 18]. Sentence-level co-occurrence has been shown to be the suitable granularity degree in relation extraction [18, 19].

In this thesis, we identify sentences that describe interactions between co-occurring biological entities using patterns containing relevant interaction between biological concepts. A pattern is defined as a sequence of specialized Part-of-Speech (POS) tags that capture the structure of key sentences in the scientific literature.

Proteins are central molecules in living systems and the description of their functions is one of the key tasks of molecular biology. Recently, the BioNLP'09 shared task [1] focused on extraction of biological events of proteins. Instead of targeting a rather simple representation of relations of proteins, e.g., protein-protein interactions, the shared task was concerned with the detailed behavior of proteins and semantically

rich biological events.

The definition of a biological event in the shared task is described as a change on the state of one or more proteins. For instance: phosphorylation of IkB involves a change on the protein IkB. Biological events are characterized by verbs or nominalized verbs that code gene and protein interactions in molecular biology articles. Compared to relations that typically deal with a pair of entities, events involve multiple participants or arguments of varying numbers. These arguments are assigned semantic roles and event types are determined based on existing biological ontologies [1], such as Gene Ontology (GO) [20] and the GENIA ontology [21]. In some cases, events are recursively embedded and function as arguments in other events, thus allowing the construction of complex conceptual networks.

In this thesis, we explore the application of the subgraph matching problem in the BioNLP field of extracting complex biological events from the scientific literature by searching for subgraphs isomorphic to the graphs of event rules within the graphs of sentences for the purpose of tackling the primary task of the BioNLP'09 shared task.

1.2 Objectives

This thesis investigates how to efficiently extract entity and relation information from the ever-growing body of scientific publications in molecular biology. The three main objectives of this thesis are:

- To achieve more accurate recognition of biological entities by employing the feature of domain-specific prefix and suffix characters that are systematically learned by an unsupervised method;

- To achieve more accurate prediction of biological interactions by selecting the text fragments that contain definite relationships between co-occurring concepts; and

- To provide an efficient way to extract complex biological events by using grammatical dependency graphs and solving the problem of pattern matching graphically.

1.3 Outline of Thesis

In Chapter 2, we review recent research on two important tasks of biological information extraction: biological entity recognition and biological relation extraction.

In Chapter 3, we propose an unsupervised method to automatically extract domain-specific prefixes and suffixes from biological corpora. The resulting affixes are then used to aid in biological entity recognition.

In Chapter 4, we propose an approach to automatically identify sentences which describe important interactions between biology concepts based on patterns that capture the structure of key sentences in the scientific literature.

In Chapter 5, we propose a graph matching-based approach to extract biological events from scientific documents by searching for subgraphs isomorphic to the graphs of event rules within the graphs of sentences for the purpose of tackling the primary task of the BioNLP'09 shared task on biological event extraction.

In Chapter 6, we conclude our work, summarize our contributions, and discuss directions for future work.

Chapter 2

Related Work

2.1 Biological Entity Recognition

An entity usually corresponds to an author's textual representation of a particular concept. It is not easy to understand an article without precise identification of entities that are used to communicate knowledge. Biological entity recognition denotes a set of procedures that are used to systematically recognize pertinent entities in the biological literature, that is, to differentiate between biological entities and non-biological entities and to highlight lexical units that are related to relevant molecular biology concepts [10, 11, 22], such as protein, DNA, RNA and cell types.

Recognizing biological entities from texts allows for text mining to capture their underlying meaning and further extract semantic relationships between entities and other useful information. Because of the importance and complexity of the problem, biological entity recognition has attracted intensive research and there is a large grow of published work on this topic [6, 8–15]. Current approaches in biological entity recognition can be generalized into three main categories: lexicon-based, rule-based and learning-based [6, 8, 9].

Lexicon-based methods use existing terminological resources, such as dictionaries or databases, in order to locate entity occurrences in texts. Given the pace of biology research, however, it is not realistic to assume that a dictionary can remain up-to-date. A drawback of lexicon-based methods is thus that they are not able to annotate all recently coined biological entities [6, 16]. In addition, a dictionary may not cover all spelling variations of biological entities in texts, so lexicon-based methods are bound to fail on the tasks that focus on locating exact entity boundaries [6, 11, 23].

Rule-based approaches attempt to detect biological entities by manually developing heuristic rules that describe various characteristics of entity formation patterns. However, rules are often time-consuming to define consistently, and are difficult to

adapt to other types of entities in the biological domain. Early on, the PROPER system [24] extracted protein names using hand-coded rules that capture surface clues of character strings in biological documents based on observation of the dataset. The system achieved high performance on a small dataset, but received a significant performance drop on another dataset that contains more entity types [16, 25]. Once a new type of entity is required to be identified, a set of rules specifically for this type has to be crafted manually. Therefore, rule-based approaches are considered to lack scalability and generalization.

Learning-based techniques use training data to learn features or patterns useful for recognizing biological entities, and have become the current trend due to the availability of well-curated biological corpora, such as the GENIA corpus [21] and the GENETAG corpus [26], in which biological entities are carefully annotated by experts. Compared to the other two methods, learning-based techniques are theoretically more capable to identify unseen or multi-word entities, and even entities with various writing styles by different authors. The JNLPBA shared task [22] is an open challenge task on recognizing biological entities in the GENIA corpus. All 8 participating systems in the JNLPBA shared task employed learning-based techniques, including Support Vector Machines (SVMs) [27], Hidden Markov Models (HMMs) [12, 14], and Conditional Random Fields (CRFs) [13, 28]. The techniques can be divided into two main approaches in terms of the ways to assign labels to individual tokens: word-based methods and sequence-based methods.

The word-based methods annotate each word without considering the previously assigned labels. The ABTA system [16] treats the entity recognition problem as a classification problem and applies a decision tree classifier to annotate entities in the biological literature. The system employs a word-by-word classification and a word n-gram model [29] is used to define each input sentence into classification instances based on a sliding window of words across sentences. Similarly, other systems employed SVMs to classify each word in texts into predefined classes [27, 30].

The sequence-based methods consider the annotation decisions on other words in order to assign a label for the current word. The best-performing system in the JNLPBA shared task employed an HMM combined with a SVM that resolves the data sparseness problem with the HMM [12]. The HMM finds the most likely labels

for words in sentences based on the assumption that the label for the current word is probabilistically dependent on all the previously assigned labels. In addition, models trained using Maximum Entropy Markov Model (MEMM) [31] and linear-chain CRFs [13] to annotate biological entities are other representative instances of sequence-based methods.

Although various approaches have been applied, a main challenge for learning-based techniques is to select a set of discriminating features that can be used for accurate annotation of biological entities. Generally, the features fall into 4 classes [11–14, 32]: (1) simple deterministic features which capture the use of uppercase letters, digits, Greek letters and other formation patterns of words, (2) morphological features such as prefix and suffix information, (3) syntactic features like Part-of-Speech (POS) tags that provide word syntactic information, and (4) semantic trigger features which capture the evidence by collecting the semantic information of key words, for instance head noun triggers or special verb triggers.

Various combinations of the features have been attempted by different systems [11, 14, 16, 32, 33]. The ABTA system [16] extracts features from each word in an n-gram instance as input for creating the classification model and classifies the word in the middle of a word n-gram. The extracted features include simple deterministic features, POS tag information, and prefix and suffix characters. Without using external domain resources, the system achieves comparable precision to state-of-the-art systems [13, 31] which resort to external knowledge to further improve the performance, such as online public databases or dictionaries. POS tag information is shown to be the most effective feature in aiding the system to annotate biological entities. This observation is also supported by other systems [14, 32] because most entities are linguistically expressed within noun phrases [7].

In contrast, for some systems [11, 33] morphological features make the most significant contribution to the performance of entity recognition. However, since there is no standard way to obtain morphological features, different approaches have been investigated [14, 16, 33]. The ABTA system [16] learns the domain-specific affixes by recording only the first and the last n characters (e.g., $n = 3$) of each word in classification instances. A similar approach is also used in [11] and the authors claimed

that the n characters could provide enough affix information for the entity recognition task. Shen *et al.* [14] and Zhou *et al.* [32] prepared 100 most frequent prefixes and suffixes from training data as candidates and evaluated them based on the difference in likelihood of part of a biological entity versus not. At a much larger scale, Kazama *et al.* [33] and Lee *et al.* [15] constructed prefix/suffix lists by sorting 10,000 prefixes/suffixes from the training data. The results show that morphological features have a substantial positive effect on improving entity annotation accuracy.

In Chapter 3, we propose an unsupervised method to automatically extract domain-specific prefixes and suffixes from biological corpora. The extracted affixes are integrated into the parametrization of the ABTA [16] system to investigate their impact on the performance of biological entity recognition. Successful demonstration of the quality of this extraction method implies that domain-specific affixes can be identified for arbitrary corpora without the need to manually generate training sets.

2.2 Biological Relation Extraction

If the biological entities are defined and localized in texts, relations among them can be inferred. Recent research in information extraction in biological science has focused on extracting semantic relations between biological entities from the scientific literature [4, 5, 34–38]. The type of relations includes protein-protein, protein-DNA, gene regulations and other interactions between macromolecules. A task of significant interest is the automated protein-protein interaction (PPI) extraction because information on relations or interactions between genes and proteins serves as a basis for generating network models of regulatory or metabolic pathways [4, 17]. Also, state-of-the-art biological entity recognition methods have achieved reasonable success, for instance a performance of 88% F-score on annotating proteins [39], which boosts the interest in biological relation extraction.

There are three main categories of methods that are proposed to extract relation information from the biological domain: machine learning techniques, pattern matching approaches, and parsing-based methods.

Machine learning techniques aim to extract biological relations by applying a set of classifiers to the test set, which is obtained by applying supervised machine learning algorithms, such as Support Vector Machines (SVMs) [40] and rule induction

algorithms [41], on the appropriately annotated training set. However, machine learning techniques were reported to be limited by the quality and extent of the training sets used to train the algorithms [42, 43]. Recently, with the introduction of corpora containing the necessary annotations [44, 45], more studies based on machine learning techniques have been conducted [40, 46, 47]. The best-performing team from the BioNLP'09 shared task on event extraction [1] used a multi-class SVM classifier incorporated with a wide array of features capturing both linear and dependency contexts to extract complex biological events [46]. Airola *et al.* extracted protein-protein interactions using a kernel-based learning algorithm in which the kernel function captures all dependency paths in the resulting dependency graphs [40, 47].

Pattern matching approaches try to first generate patterns or rules which model classes of the underlying biological relations and then use the generated patterns to detect specific relations. The patterns can be either simple rules defined for extracting general relations, or complicated rules for discovering special interactions. Therefore, pattern matching approaches are able to extract relations from long or complex sentences. Also, patterns or rules are usually developed by domain experts which provide an advantage in evaluation [17] but is a problem to keep current in the long term. In the "KDD Challenge Cup" [48] for curating biological databases, the participating systems which applied pattern matching approaches generally outperformed systems using other strategies to extract biological relations.

Early on, Blaschke [49] employed patterns to predict the presence of a protein-protein interaction. A series of patterns was developed manually to cover the common descriptions of protein functions. This process was based on a set of keywords, including interaction verbs, that are commonly used to describe this type of interaction. A sentence extraction system BioIE [50] also used patterns to extract entire sentences related to protein families, protein structures, functions and diseases. The patterns were manually defined and consisted of single words, word pairs, and small phrases. In GENIES, Friedman and colleagues [34] manually developed more complex patterns with both syntactic and semantic constraints to extract and structure information related to molecular pathways.

Although systems relying on hand-coded patterns [34, 35, 49, 50] achieved some success in extracting biological interactions, the strict requirement of dedicated expert

work is problematic. Moreover, each type of interaction may require a definition of many different patterns including different arrangements and different variants of the same keyword. Manually encoding all patterns encountered in a corpus is time-consuming and potentially impractical in real applications. Therefore, automatically learning such patterns is a more practical solution to maintain such resource in the long term.

An approach which combines dynamic programming and biological sequence alignment algorithms [51] was introduced by Huang *et al.* [36]. This approach is designed to generate patterns useful for extracting protein-protein interactions. The main problem with this approach is that the alignment algorithm relies on an alignment scoring scheme which contains a number of free parameters. We have showed that finding the optimal scoring scheme of parameters for edit distance-based sequence classification is an NP-hard problem [52, 53].

In Chapter 4, we propose a novel approach to automatically extract sentence patterns that contain interactions involving concepts of molecular biology. The patterns are then used to identify sentences that contain important relationships between biological concepts. A pattern is defined as a sequence of specialized Part-of-Speech (POS) tags that capture the structure of key sentences in the scientific literature. Each candidate sentence for the classification task is encoded as a POS array and then aligned to a collection of pre-extracted patterns. The quality of the alignment is expressed as a pairwise alignment score. The most innovative component of this work is the use of a Genetic Algorithm (GA) to maximize the classification performance of the alignment scoring scheme.

GAs were also used as a learning strategy to train finite state automata for finding biological relation patterns in texts as well as to optimize the extracted patterns [54]. In addition, phrasal patterns that describe protein functions were mined from sentences via a strict sentence pattern mining mechanism and patterns were then matched with new texts to recognize descriptions of protein functions in articles [55]. It was reported [54, 56–58] that automatically learned patterns identify biological interactions even more accurately than hand-coded patterns.

Pattern matching approaches can cope with long or complex sentences, however, the linear surface patterns captured by shallow analysis on the biological texts can

not process the sentences that have long-range dependencies involving multiple biological entities or biological processes [42]. In contrast, **parsing-based methods** can discover biological relation information from such sentences by determining syntactic relationships between words. In the GENIA event corpus [44], the distances among proteins and other arguments of biological events in terms of shortest dependency path length are considerably shorter than in terms of their linear order in the sentence [46].

A study to evaluate four state-of-the-art extraction methods of gene and/or protein interactions was conducted on five publicly available corpora of biological interactions [59]. The method, AkanePPI [60], combined with a deep syntactic parser achieved the best overall performance. In practice, full parsing is now the mainstream approach and the basis for biological relation extraction tasks [1, 4–6, 34, 35, 37, 61]. It is reported that the accuracy for parsing of biological text is now in the 80-90% range [62, 63]. Recently, a comparative evaluation was performed on eight representative natural language parsers from different parsing frameworks, focusing on the task of PPI extraction from biological papers [64]. The contributions of each parser with multiple parse representations to PPI extraction were evaluated thoroughly by reporting the relationship between the size of the dataset used to train the parser, the parser's accuracy and the overall PPI extraction accuracy when the parser is used as a component.

In Chapter 5, we explore the application of the subgraph matching problem in the BioNLP field for extracting complex biological events from the scientific literature. The approach to extract biological events that we propose here belongs to the category of parsing-based methods. The approach may also be generalized to extract events from other domains where training data is available as it requires neither manual intervention or external domain-specific resources.

Chapter 3

An Unsupervised Method for Extracting Domain-specific Affixes in Biological Literature

In this chapter, we propose a method to automatically extract domain-specific prefix and suffix characters from biological corpora. As one application of the proposed method, the resulting affix characters are integrated into the parametrization of the ABTA [16] system to investigate their impact on the performance of entity recognition. The proposed method is completely unsupervised. For this reason, we suggest that our method can be generalized for extracting domain-specific affixes from many domains.

The chapter is organized as follows: In section 3.1, we introduce the problem of affix extraction in the task of biological entity recognition. Section 3.2 elaborates the methodology that we propose for affix extraction. The experiment results are presented and evaluated in section 3.3. Finally, section 3.4 summarizes this work and introduces future directions.

3.1 Problem Description

Jiampojamarn *et al.* [16] proposed the ABTA system which applies supervised learning methods to annotate biological entities in the scientific literature. Given unstructured texts in biological research, the annotation system first identifies biological entities based on five word position classes, "Start", "Middle", "End", "Single" and "Non-relevant", indicating the position of each word within a biological entity. Therefore, multi-word biological entities should be in a consistent sequence of classes "Start (Middle)* End" while single word entities will be indicated by the class "Single". The system then classifies the identified entities into biological concepts: proteins, DNA, RNA, cell types and cell lines. The system employs a word-by-word classification and a word n-gram model [29] is used to define each input sentence into instances to be classified. The word n-gram model is a fixed window size of n words sliding from the

beginning to the end of each sentence. The system extracts features from each word in an n-gram as input for creating the classification model and classifies the word in the middle of a word n-gram. The extracted features include simple deterministic features, POS tag information, and prefix and suffix characters.

The ABTA system [16] learns the domain-specific affixes by recording only the first and the last n characters (e.g., $n = 3$) of each word in classification instances. A similar approach is also adopted in [11] and the authors claimed that the n characters could provide enough affix information for the entity recognition task.

However, it is often observed that most biological entities tend to employ longer affixes which carry specific semantic meanings about the entities. For instance, the biological entity "EBNA2-oestrogen" contains a typical prefix "EBNA2-" while another biological entity "CD8(+)-T-cell" employs a representative suffix "-T-cell". Using a specific number of characters to provide affix information could lose important affixes or generate inaccurate affixes in these cases. It is more likely that a specific list of typically used prefixes and suffixes of biological words would provide more accurate information to classifying some biological entities and boundaries. Therefore, we hypothesize that a more flexible affix definition will improve the performance of the task of biological entity recognition.

Shen *et al.* [14] explored an adaptation of a general Hidden Markov Model-based entity recognizer to the biological domain. They experimented with POS tags, prefix and suffix information and noun heads as features. 100 most frequent prefixes and suffixes were extracted from training data as candidates, and evaluated based on difference in likelihood of part of a biological entity versus not. Two limitations of this affix extraction method are: (1) use of only a biological corpus, so that the general domain-independent affixes are not removed, and (2) a supervised process of choosing a score threshold that is used in affix selection.

3.2 PATRICIA-Tree-based Affix Extraction

3.2.1 PATRICIA Trees

The method we propose to extract affixes from biological words is based on the use of PATRICIA trees. "PATRICIA" stands for "Practical Algorithm To Retrieve

Information Coded In Alphanumeric". It was first proposed as an algorithm to provide a flexible means of storing, indexing, and retrieving information in a large file [65]. A PATRICIA tree uses path compression by grouping common sequences into nodes. This structure provides an efficient way of storing values while maintaining the lookup time for a key of $O(N)$ in the worst case, where N is the length of the longest key. Because of the outstanding flexibility and efficiency, PATRICIA trees have been applied to many large information retrieval problems [66, 67].

There is a precedent for the use of PATRICIA trees to perform an automatic extraction of keywords from relevant Chinese documents [66]. The tree structure helps to reduce the difficulty in processing Chinese language which is known to lack explicit word boundaries. Refined by a mutual information-based algorithm and a general-word lexicon, the extracted keywords have proved useful in several Chinese information retrieval applications, such as book indexing, document classification and relevance feedback. A learning device based on PATRICIA trees was also developed to efficiently store and retrieve important information [67]. The original tree structure was enhanced with a deletion mechanism which allowed memory to be released by automatically deleting unimportant information for the purpose of storing new inputs. This enhanced PATRICIA tree has the ability to accept unlimited amounts of information and the potential to be an online training tool.

Figure 3.1 illustrates a simple example of the growth of a PATRICIA tree under a sequence of insertions. Suppose "ababb", "ababa", "ba" and "aaabba" are four words to be inserted into PATRICIA tree. Initially, the tree is empty. The word "ababb" is first inserted as a node of the tree. In order to insert the next word "ababa", it is necessary to compare it with the existing node containing the word "ababb". The only difference of the two words is the fifth character. Thus, the common substring "abab" is split out from both words to form a new node, while the remaining characters of each word, "a" and "b", become two descendant nodes of this node. Similarly, it is necessary to compare the next inserted word "ba" with "ababb". As they are different from the first character, a new node for "ba" is then created from the root of the tree. Finally, since the common character of the last inserted word "aaabba" and "ababb" is "a", it is then split out as a new node.

In our work, all biological words are inserted and stored in a PATRICIA tree,

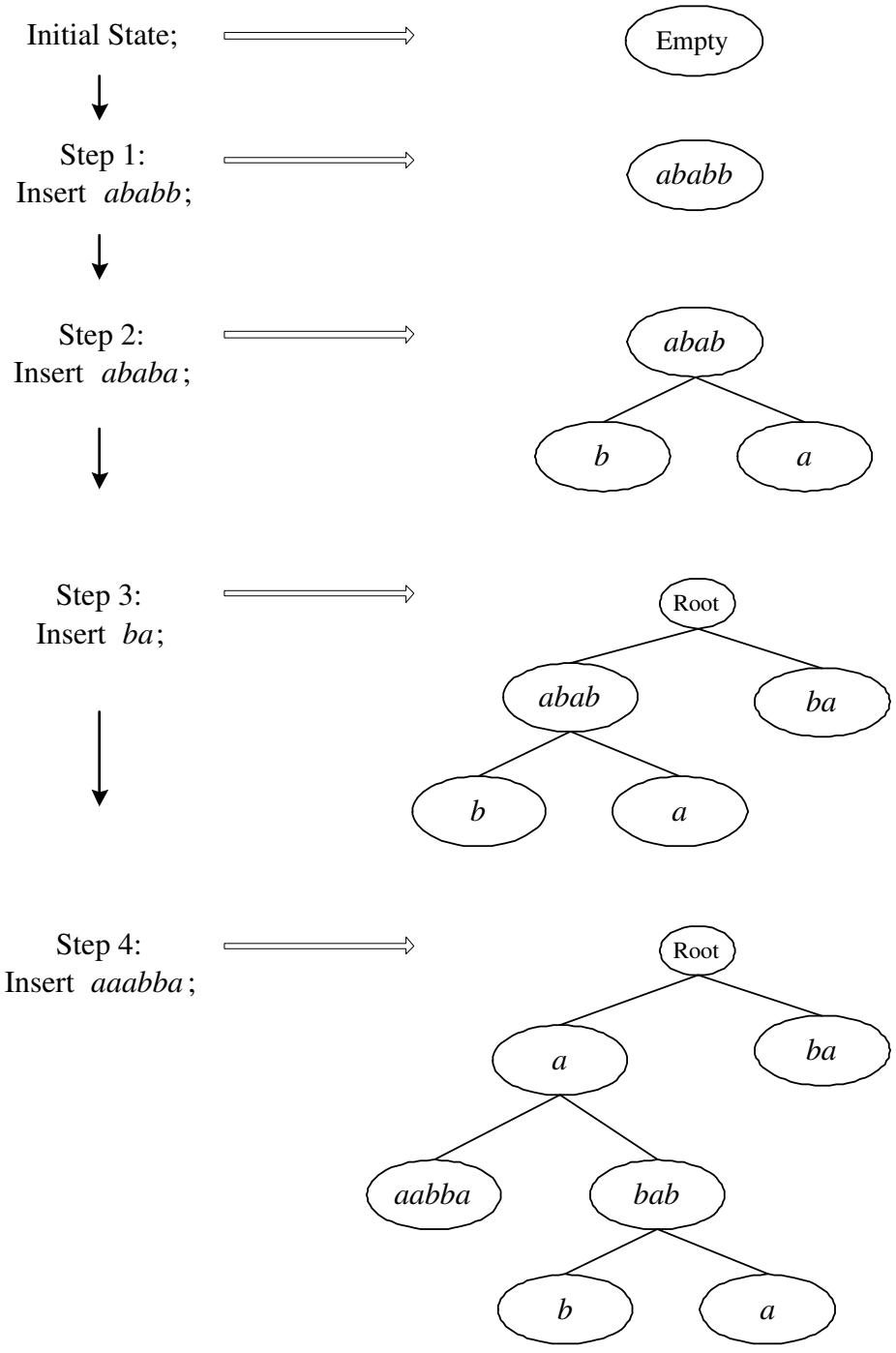


Figure 3.1: Growth of a PATRICIA Tree

which can be used to efficiently look up specific biological words, extract biological words that share specified affixes and calculate required statistics.

3.2.2 Terminology

In order to clearly describe our proposed method, it is necessary to introduce and clarify some terminologies used in the process of affix extraction.

We assume that text is composed of characters from a finite alphabet. The *alphabet* is a finite set of symbols Σ. In practice, elements of the alphabet are ASCII characters. In our further discussion, we will assume that the alphabet is given at the beginning and always the same, unless specified differently. A *string* is a finite sequence of characters from Σ, which can be empty. The *length of a string* w is the number of characters in the sequence w, and it is denoted $|w|$. An *empty string*, i.e., an empty sequence of characters, is denoted ϵ and its length is zero: $|\epsilon| = 0$.

A *text,* which is in practice typically a scientific article, is simply a long string. Two strings s and t can be concatenated by appending one string to another, which is denoted as st or $s \cdot t$. This operation is called *concatenation.* The length of concatenated string is the sum of length of operand strings; i.e., $|st| = |s| + |t|$. The empty string acts as the neutral element with respect to the concatenation operation; i.e., for any string s, $s\epsilon = \epsilon s = s$. We say that string s is a *substring* of string t if $t = xsy$ for some strings x and y.

The standard definition of *prefix* and *suffix* in the formal languages theory is that a string s is a prefix of string t if $t = sx$ for some string x, and s is a suffix of string t if $t = xs$ for some string x. However, since we are going to define terms *prefix* and *suffix* in a different way in our proposed method, we will call these standard notions of *prefix* and *suffix*, *general prefix* and *general suffix,* or if we want to include both terms under one term, *general affix.* Throughout our work, the word *affix* will mean *prefix* or *suffix.*

We define *words* as the maximal substrings of a text consisting only of the characters from a distinguished subset of Σ, which is the set of all letters, digits, and some special signs such as the hyphen sign '-' and the slash sign '/'. A *corpus* (*pl. corpora*) is a finite set of texts. We say that a word w occurs in a corpus $\mathcal{C}$ if there is a text $T \in \mathcal{C}$ such that w occurs in T.

The goal of our method is to extract a set of domain-specific affixes (i.e., prefixes and suffixes), so that we can use them as a special feature in biological entity detection. Being a *domain-specific affix* is not a deterministic feature of a string, however, we use a probabilistic model next to predict a probability that given an arbitrary occurrence of an affix in a text, it has a domain-specific function, i.e., its occurrence is related to the biological domain in some way. The task becomes more difficult by performing this function in a very unsupervised way in the sense that we do not use information about annotated biological entities, but treat the domain-specific corpus as unmarked, uniform text.

The first step is to identify potential affixes, which we define in the following way:

Definition 3.1. (Potential Affix) Given a corpus $\mathcal{C}$ over an alphabet Σ, a string p, $|p| \geq 2$, is a *potential prefix* if there are two distinct words w_1 and w_2 occurring in $\mathcal{C}$, such that $w_1 = px_1$ and $w_2 = px_2$ for two strings x_1 and x_2. Similarly, a string s, $|s| \geq 2$, is a *potential suffix* if there are two distinct words w_1 and w_2 occurring in $\mathcal{C}$, such that $w_1 = x_1 s$ and $w_2 = x_2 s$ for two strings x_1 and x_2. A *potential affix* is a potential prefix or potential suffix.

It is seen in Figure 3.1 that every node in PATRICIA tree contains exactly one string of 1 or more characters, which is the preceding substring of its descendant nodes. Meanwhile, every word is a path of substrings from the root to a leaf. An efficient algorithmic way of finding all potential prefixes is to collect all words of a corpus in a PATRICIA tree, and then by concatenating all strings in the nodes of all paths from the root to any other internal node in the tree we obtain potential prefixes. The prefixes of length 1 are excluded, since very short prefixes and suffixes occur too frequently to be useful. The potential suffixes are extracted in a similar way by reversing all words before adding them to the tree, and then by reversing the extracted potential prefixes at the end.

Next, affixes are extracted from potential affixes. The definition for a prefix, suffix, and affix adopted in our proposed method is given as follows, and will be described in more detail while introducing our probabilistic model.

Definition 3.2. (Affix) A string p is a *prefix* in a corpus $\mathcal{C}$ if there are two words w_1 and w_2 occurring in the corpus $\mathcal{C}$ such that $w_1 = pw_2$. We say that p *occurs as a prefix* in word w_1. A string s is a *suffix* in corpus $\mathcal{C}$ if there are a word w_1 and a word or prefix w_2 occurring in the corpus $\mathcal{C}$ such that $w_1 = w_2 s$. We say that s *occurs as a suffix* in word w_1. *Affix* is prefix or suffix.

3.2.3 Methodology

In this work, we have designed the experiments to extract domain-specific prefixes and suffixes of biological words from a biological corpus, and investigate the impact of the extracted affix information on the performance of biological entity recognition. The overall design of our experiments consists of three major processes: affix extraction, affix refining and evaluation of experimental results.

In the affix extraction process, we first populate a PATRICIA tree using all words in the combined corpus*(CC)* of a Biological Corpus *(BC)* and a General English Corpus *(GEC)*. *GEC* is used against *BC* in order to extract more accurate biological affix information. Two PATRICIA trees are populated separately for extracting prefixes and suffixes. The suffix tree is based on strings derived by reversing all the input words from the combined corpus. According to Definition 3.1, all the potential prefixes and suffixes are then extracted from the populated PATRICIA trees.

In the affix refining process, for each extracted potential affix, we compute its joint probability of being both an English affix and a biological affix, $P(D = \mathit{Biology}, A = \mathit{Yes}|PA)$, where D stands for *Domain*, A stands for *Affix* and PA represents *Potential Affix*. This joint probability can be further decomposed as shown in Eq.(3.1). In the formula, $P\,(A = \mathit{Yes}|PA)$ denotes the probability that a given potential affix is a true English affix while $P(D = \mathit{Biology}|A = \mathit{Yes}, PA)$ refers to the probability that a given English affix is actually a biological affix.

$$\begin{aligned} &P(D = \mathit{Biology}, A = \mathit{Yes}|PA) = \\ &\quad P(D{=}\mathit{Biology}|A{=}\mathit{Yes}, PA) \times P(A{=}\mathit{Yes}|PA) \end{aligned} \tag{3.1}$$

To calculate $P\,(A = \mathit{Yes}|PA)$, the probabilities of prefixes and suffixes are measured separately. According to Definition 3.2, a prefix can be found by enumerating,

for each node, all descendant substrings and assessing their existence as stand-alone words. Using some domain-specific words as example, "radioimmunoassay", "radioiodine" and "radiolabeled" have a common starting string "radio". If we take out the remaining part of each word, three new strings are obtained, "immunoassay", "iodine" and "labeled". Since all the input words are already stored in PATRICIA tree, we look up these three strings in PATRICIA tree and find that "immunoassay", "iodine" and "labeled" are also meaningful words in the tree. This indicates that "radio" is a prefix among the input words. On the other hand, it is obvious that "radioimmunoassay" and "radioiodine" share another string "radioi". However, "mmunoassay" and "odine" are not meaningful words due to their absence in the PATRICIA tree. This suggests that "radioi" is not a prefix.

For each extracted potential prefix, $P\left(A = Yes|PA\right)$ is computed as the proportion of strings formed by traversing all descendant nodes that are meaningful words. In our experiments, the measure of assessing if a string is meaningful is to look up whether each string is an existing word present in the built prefix PATRICIA tree. Algorithm 1 shows the procedure of populating a PATRICIA tree and calculating $P\left(A = Yes|PA\right)$ for each potential prefix. A time complexity of $O(n)$ is estimated for the algorithm and n is the size of the combined corpus.

Likewise, a suffix can be also detected in terms of Definition 3.2. Similar to the idea of calculating $P\left(A = Yes|PA\right)$ for potential prefix, we conjecture that the extracted potential suffix could be a reasonable English suffix if the inverted strings formed from traversing the descendant nodes of the potential suffix in the suffix PATRICIA tree are meaningful words. For instance, "Calcium-dependent", "Erythropoietin-dependent" and "Ligand-dependent" share a common ending string "-dependent". Since the remaining strings of each word, "Calcium", "Erythropoietin" and "Ligand" can be found in the "forward" PATRICIA tree, "-dependent" is a potentially useful suffix.

However, it is often observable that some English words do not begin with another word but a meaningful prefix, for example, "pre-mRNA" and "pro-glutathione". It is known that "-mRNA"and "-glutathione" are good suffixes in biology. "pre" and "pro", however, are not typical words but meaningful prefixes, and in fact have been extracted when calculating $P\left(A = Yes|PA\right)$ for potential prefix. Therefore, in order

Algorithm 1 $P\,(A = \mathit{Yes}|PA)$ for Prefix

Input: words $(w) \in$ Combined Corpus (CC)

Output: $P\,(A = \mathit{Yes}|PA)$ for each potential prefix

```
1:  PT = ∅                                                    //PT : Patricia Trie
2:  for all words w ∈ CC do
3:      PT ← Insert(w)                                        //Populating Patricia Trie
4:  for all nodes n_i ∈ PT do
5:      PA ← String(n_i)   //Concatenate strings in nodes from root to n_i, which is a
6:      //potential prefix
7:      T_PA ← PrefixSearch(PA)          //PrefixSearch() returns all words w ∈ CC
8:      //beginning with PA
9:      score ← 0
10:     for all words w ∈ T_PA  do
11:         if  Extrstr(PA, w) in PT  then
12:             //Extrstr() returns the remaining string of w without PA
13:             score ++
14:     P(A = Yes|PA) ← score/|T_PA|
15:     //|T_PA| is the number of words in T_PA
```

to detect and capture such potential suffixes, we further assume that if a word begins with a recognized prefix instead of another typical word, the remaining part of the word still has the potential to be an informative suffix. Therefore, strings "-mRNA" and "-glutathione" can be successfully extracted as potential suffixes. Based on our probability model, an extracted potential prefix is considered a recognized prefix in our experiments if its $P\,(A = \mathit{Yes}|PA)$ is greater than 0.5.

To calculate $P(D = \mathit{Biology}|A = \mathit{Yes}, PA)$, it is necessary to first determine true English affixes from extracted potential affixes. In our experiments, we consider that an extracted potential prefix or suffix is a recognized affix only if its $P\,(A = \mathit{Yes}|PA)$ is greater than 0.5. It is also necessary to consider the biological corpus BC and the general English corpus GEC separately. It is assumed that a biology-related affix tends to occur more frequently in words of BC than GEC. Eq.(3.2) is used to estimate $P\,(D = \mathit{Biology}|A = \mathit{Yes}, PA)$.

$$
\begin{aligned}
&P\left(D = \textit{Biology} | A = \textit{Yes}, PA\right) = \\
&\quad \left(\textit{Number of Words with } PA \textit{ in } BC / \textit{Size}\left(BC\right)\right) / \\
&\quad \left(\textit{Number of Words with } PA \textit{ in } BC / \textit{Size}\left(BC\right) + \right. \\
&\left. \textit{Number of Words with } PA \textit{ in } GEC / \textit{Size}\left(GEC\right)\right), \qquad (3.2)
\end{aligned}
$$

where only PA with $P\left(A = \textit{Yes} | PA\right)$ greater than 0.5 are used, and the number of words with a certain PA is further normalized by the size of each corpus.

Finally, the joint probability of each potential affix, $P\left(D = \textit{Biology}, A = \textit{Yes} | PA\right)$, can be used to parametrize a word beginning or ending with PA.

In the evaluation process of our experiments, the prefix-suffix pair with maximum joint probability values is used to parametrize a word. Therefore, each word in BC has exactly two values as affix feature: a joint probability value for its potential prefix and a joint probability value for its potential suffix. We then replace the original affix feature of ABTA system with our obtained joint probability values, and investigate whether these new affix information leads to equivalent or better entity recognition on BC.

3.3 Results and Evaluation

3.3.1 Dataset and Environment

For our experiments, it is necessary to use a corpus that includes widely used biological entities and common English words. This dataset, therefore, will allow us to accurately extract the information of biology related affixes. As a proof-of-concept prototype, our experiments were conducted on two widely used corpora: GENIA corpus (v3.02) [21] and Brown corpus [68]. The GENIA version 3.02 corpus is used as the biological corpus BC in our experiments. It contains 2,000 biological research paper abstracts. They were selected from the search results in the MEDLINE database [2], and each biological entity has been annotated into different terminal classes based on the opinions of experts in biology. Used as the general English corpus GEC, Brown corpus consists of 500 text samples of common English usage, drawn from 15 different text categories, totaling about a million words.

All the experiments were executed on a Sun Solaris server Sun-Fire-880. Our experiments were implemented using Perl and Python.

3.3.2 Experimental Results

We extracted 15,718 potential prefixes and 21,282 potential suffixes from the combined corpus of GENIA and Brown. Among them, there are 2,306 potential prefixes and 1,913 potential suffixes with joint probability value $P\,(D = \mathit{Biology}, A = \mathit{Yes}|PA)$ greater than 0.5. Table 3.1 shows a few examples of extracted potential affixes whose joint probability value is equal to 1.0. It is seen that most of these potential affixes are understandable biological affixes which directly carry specific semantic meanings about certain biological entities. However, some substrings are also captured as potential affixes although they may not be recognized as "affixes" in linguistics, for example "adenomyo" in prefixes, and "plasias" in suffixes. In the GENIA corpus, "adenomyo" is the common beginning substring of biological entities "adenomyoma", "adenomyosis" and "adenomyotic" , while "plasias" is the common ending substring of biological entities "neoplasias" and "hyperplasias". The whole list of extracted potential affixes is available upon request.

Potential Prefixes			
13-acetate	adenomyo	3-kinase	platelet
B-cell	Rel/NF-kappaB	CD28	pharmaco
endotoxin	anti-CD28	HSV-1	adenovirus
I-kappaB	VitD3	ligand	chromatin
macrophage	cytokine	N-alpha-tosyl-L	hemoglobin
Potential Suffixes			
-T-cell	-alpha-activated	cytoid	-methyl
-coated	mopoiesis	-bearing	lyse
-expressed	-nonresponsive	-kappaB-mediated	-receptor
-inducer	coagulant	-globin-encoding	glycemia
plasias	-soluble	-immortalized	racrine

Table 3.1: Examples of extracted potential affixes with joint probability value 1.0

In order to investigate whether the extracted affixes improve the performance of biological entity recognition, it is necessary to obtain the experimental results of both the original ABTA system and the ABTA system using our extracted affix

information. In ABTA, the extraction of features is performed on the whole 2000 abstracts of the GENIA corpus, and then 1800 abstracts are used as training set while the rest 200 abstracts are used as testing set.

The evaluation measures are precision, recall, F-score and classification accuracy. Precision measures the proportion of the number of correctly recognized words in a class to all the recognized words in the class. Recall calculates the ratio of the number of correctly recognized words in a class to all the correct words in the class. F-score is a weighted harmonic mean of recall and precision, which are given equal importance in this work. Classification accuracy measures the proportion of the number of correctly classified words in all classes to the total number of words of all classes.

C4.5 decision tree classifier [69] is reported as the most efficient classifier which leads to the best performance among all the classifiers experimented in [16]. Therefore, C4.5 is used as the main classifier in our experiments. The experimental results of the ABTA system with 10-fold cross-validation based on different combinations of the original features are presented in Table 3.2 according to the five word position classes, "Start", "Middle", "End", "Single" and "Non-relevant", indicating the position of each word within a biological entity. *"SD"* is short for Simple Deterministic features, *"AC"* denotes Affix Characters, and *"POS"* refers to POS tag information obtained by a statistical tagger Lingua::EN::Tagger [70]. The setting of parameters in the experiments with ABTA is: the word n-gram size is 3, the number of word feature patterns is 3, and the number of affix characters is 4. We have reported the F-score and the classification accuracy of the experiments in the table. It is seen that there is a tendency with the experimental performance that for a multi-word biological entity, the middle position is most difficult to detect while the ending position is generally easier to be identified than the starting position. The assumed reason for this tendency is that for multi-word biological entities, many middle words are seemingly unrelated to biology domain while many ending words directly indicate their identity, for instances, "receptor", "virus" and "expression".

Table 3.3 shows the experimental results of the ABTA system after replacing the original affix feature with our obtained joint probability values for each word in the GENIA corpus. *"JP"* is used to denote Joint Probability values. It is seen that based on all three features the system achieves a classification accuracy of 87.5%, which

Feature sets	F-score					Classification Accuracy (%)	# Parameters
	Start	Middle	End	Single	Non		
SD	0.467	0.279	0.495	0.491	0.864	74.59	9
AC	0.709	0.663	0.758	0.719	0.932	85.67	24
POS	0.69	0.702	0.775	0.67	0.908	83.96	3
SD+AC	0.717	0.674	0.762	0.730	0.933	86.02	33
SD+POS	0.726	0.721	0.793	0.716	0.923	85.96	12
AC+POS	0.755	0.741	0.809	0.732	0.930	87.14	27
SD+AC+POS	0.764	0.745	0.811	0.749	0.933	**87.59**	**36**

Table 3.2: Performance of original ABTA system with respect to features

is comparable to the results of the original ABTA system. However, the size of the feature set of the system is significantly reduced, and the classification accuracy of 87.5% is achieved based on only 18 parameters, which is 1/2 of the size of the original feature set. Meanwhile, the execution time of the experiments generally reduces to nearly half of the original ABTA system (e.g., reduces from 4 hours to 1.7 hours). Furthermore, when the feature set contains only our extracted affix information, the system reaches a classification accuracy of 81.46% based on only 6 parameters. It is comparable with the classification accuracy achieved by using only POS information in the system. In addition, Table 3.3 also presents the experimental results when our extracted affix information is used as an additional feature to the original feature set. It is expected that the system performance is further improved when the four features are applied together. However, the size of the feature set increases to 42 parameters, which increases the data redundancy. This proves that the extracted affix information has a positive impact on locating biological entities, and it could be a good replacement of the original affix feature.

Feature sets	F-score					Classification Accuracy (%)	# Param.
	Start	Middle	End	Single	Non		
JP	0.652	0.605	0.713	0.602	0.898	**81.46**	**6**
SD+JP	0.708	0.680	0.756	0.699	0.919	84.84	15
JP+POS	0.753	0.740	0.805	0.722	0.928	86.92	9
SD+JP+POS	0.758	0.749	0.809	0.74	0.933	**87.50**	**18**
SD+AC+POS+JP	0.767	0.746	0.816	0.751	0.934	87.77	42

Table 3.3: Performance of ABTA system with extracted affix information with respect to features

Moreover, we also evaluated the performance of the exact matching of biological entity recognition based on the obtained experimental results of the ABTA system. The exact matching annotation in the ABTA system is to accurately identify every biological entity, including both multi-word entities and single word entities, therefore, all the word position classes of an entity have to be classified correctly at the same time. An error occurring in any one of "Start" "Middle" and "End" classes leads the system to annotate multi-word entities incorrectly. Consequently, the accumulated errors will influence the exact matching annotation performance. Table 3.4 presents the best obtained exact matching annotation results of different combination of features based on 10-fold cross-validation over the GENIA corpus. It is seen that after replacing the original affix feature of the ABTA system with our obtained joint probability values for each word in the GENIA corpus, the system achieves an 0.702 F-score on exact matching of biological entity recognition, comparable to the exact matching performance of the original ABTA system. In addition, when the feature set contains only our extracted affix information, the system reaches a 0.583 F-score on exact matching. Although it is a little lower than the exact matching performance achieved by using only the original affix features in the system, the feature set size of the system is significantly reduced from 24 to 6.

Feature sets	Exact Matching Annotation			# Parameters
	Precision	Recall	F-score	
AC	0.637	0.577	0.606	24
SD+AC+POS	0.748	0.666	0.705	36
JP	0.618	0.552	0.583	6
SD+JP+POS	0.745	0.663	0.702	18

Table 3.4: Exact matching annotation performance with respect to features

In order to further compare our method with the original ABTA system, we attempted eleven different sizes of training data set to run the experiments separately based on our method and the original ABTA system. They can then be evaluated in terms of their performance on each training set size. These eleven different training set sizes are: 0.25%, 0.5%, 1%, 2.5%, 5%, 7.5%, 10%, 25%, 50%, 75% and 90%. For instance, 0.25% denotes that the training data set is 0.25% of the GENIA corpus [21] while the rest 99.75% becomes the testing data set for experiments. It is observed

that there are about 21 paper abstracts in training set when its size is 1% , and 52 abstracts when its size is 2.5%.

For each training set size, we randomly extracted 10 different training sets from the GENIA corpus to run the experiments. We then computed the mean F-score of 10 obtained overall F-scores for the exact matching of biological entity recognition. Figure 3.2 is drawn to illustrate the distribution of mean F-score of each training set size for both methods, with the incremental proportion of training data. The X axis in Figure 3.2 has been log-scaled with base 10 in order to better compare the results of two methods. It is noted that the change patterns of mean F-score obtained by our method and the original ABTA system are similar. It is also seen that our method achieves marginally better annotation performance when the proportion of training data is under 2.5%.

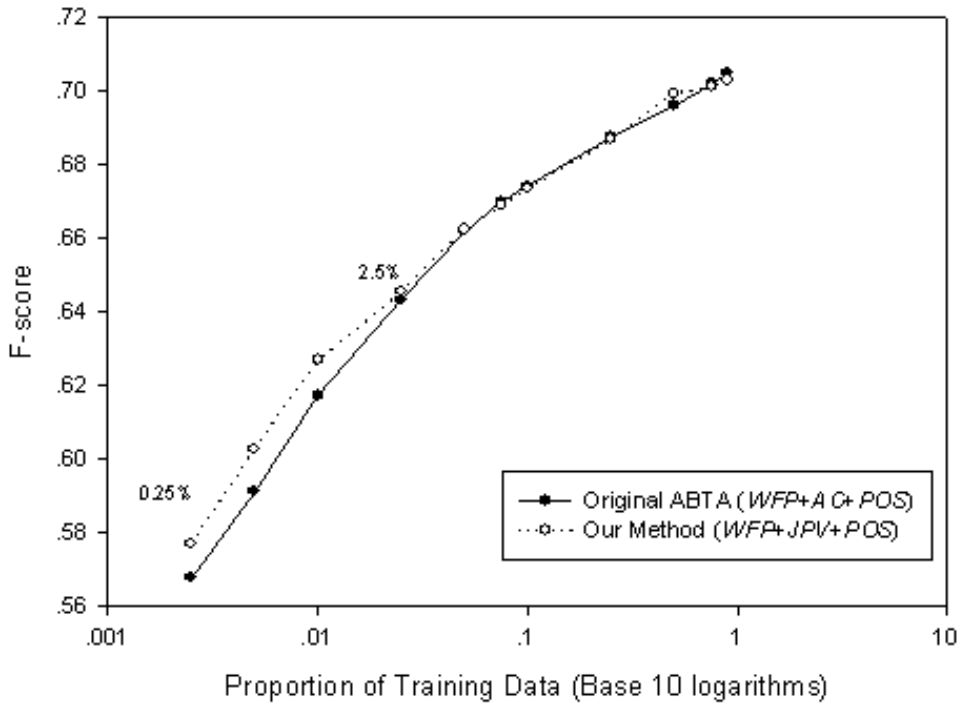


Figure 3.2: Distribution of Mean F-score

In order to determine if the annotation performance difference between our method and the original ABTA system is statistically significant, we performed one-tailed t-Test [69] on the annotation results with our hypothesis that the mean F-score of our proposed method is higher than the mean F-score of the original ABTA system.

The level of significance α is set to be the conventional value 0.05. As a result, the annotation performance difference between two methods is statistically significant when the proportion of training data is 0.25%, 0.5%, 1% or 2.5%. Table 3.5 shows the P values of t-Test results for the various training set sizes. This demonstrates that the ABTA system adopting our method outperforms the original ABTA system when the proportion of training data is lower than 2.5% of the GENIA corpus, and achieves comparable performance with the original ABTA system when the proportion continuouly increases.

A pre-annotated data set is not always available for some domains, for example the biological domain. Therefore, it is necessary for domain experts to manually annotate the raw data sets. Instead of annotating a large data set by hand, it is understandable that domain experts are more willing to manually evaluate small amount of data. The annotated data can then be used as a seed training data set for the annotation system to further annotate the experimental data. Hence, our method would be more preferable in this case as it performs better with small amount of training data.

One-tailed t-Test	Training set size			
	0.25%	0.5%	1%	2.5%
P value	0.0298	0.0006	0.0002	0.0229

Table 3.5: One-tailed t-Test results

Since the original ABTA system [16] was also evaluated on the JNLPBA shared task of biological entity recognition [22] by classifying identified entities into biological concept classes, we further applied our modified ABTA system on the JNLPBA dataset. The training set contains 2,000 Medline abstracts labeled with biological classes in the "IOB" style, which utilizes three types of tags: <B> for the beginning word of an entity, <I> for the remaining words of an entity and <O> for non-entity words. The testing set is composed of 404 new Medline abstracts. The performances of the original ABTA system and the ABTA system adopting our method on the testing set with respect to each biological concept class are given in Table 3.6 and Table 3.7 respectively. It is observed that the ABTA system adopting our method outperforms the original ABTA system in identifying proteins, DNA and RNA but obtains lower performance in discovering cell types and cell lines. This suggests that the extracted affix information tends to be more decisive for protein, DNA and RNA

entities due to their characteristic morphology.

Class (# of entities)	Precision	Recall	F-score
Protein (5,067)	0.663	0.596	0.628
DNA (1,056)	0.647	0.509	0.570
RNA (118)	0.616	0.340	0.438
Cell_type (1,921)	0.566	0.513	0.538
Cell_line (500)	0.410	0.361	0.384
All classes	0.649	0.526	0.581

Table 3.6: Performance of original ABTA system on testing set of JNLPBA

Class (# of entities)	Precision	Recall	F-score
Protein (5,067)	0.670	0.594	0.630
DNA (1,056)	0.653	0.507	0.571
RNA (118)	0.622	0.341	0.441
Cell_type (1,921)	0.548	0.501	0.523
Cell_line (500)	0.397	0.353	0.374
All classes	0.652	0.522	0.580

Table 3.7: Performance of ABTA system with extracted affix information on testing set of JNLPBA

In addition, Table 3.8 presents a comparison of our results with the results of the top four participating systems in the JNLPBA shared task, which are taken from the task report [22]. The table also includes the baseline system provided for the task, which is based on the longest string matching against a list of entities from the training data. Without using any dictionary or other external domain resources, the original ABTA system and our modified ABTA system achieve comparable precision with those participating systems, and the overall performances in F-score are all much higher than the baseline system. It is also obvious that our modified ABTA system obtains a better precision compared to the original ABTA system. However, the less performance in recall leads to a comparable F-score in overall performance.

3.4 Summary

In this chapter, we have presented an unsupervised method to extract domain-specific prefixes and suffixes from a biological corpus based on the use of PATRICIA tree.

System	Precision	Recall	F-score
Zhou and Su [12]	0.694	0.760	0.726
Finkel [31]	0.686	0.716	0.701
Settles [13]	0.693	0.703	0.698
Song [28]	0.648	0.678	0.663
Original ABTA [16]	0.649	0.526	0.581
Our Modified ABTA	**0.652**	**0.522**	**0.580**
Baseline [22]	0.476	0.508	0.491

Table 3.8: Performance comparison with respect to JNLPBA task

As one application of our proposed method, the ABTA system [16] adopting our method achieves an overall classification accuracy of 87.5% in locating biological entities, and derives an 0.702 F-score in exact entity matching annotation, which are all comparable to the experimental results obtained by the original ABTA system. However, our method helps the system significantly reduce the size of feature set and thus improves the system efficiency. The system also obtains a classification accuracy of 81.46% based only on our extracted affix information. This demonstrates that the affix information achieved by the proposed method is important to accurately locating biological entities. The evaluation on the JNLPBA shared task suggests that the extracted affix information is more sensitive in identifying proteins, DNA and RNA.

We further explored the reliability of our method by gradually increasing the proportion of training data from 0.25% to 90% of the GENIA corpus. One-tailed t-Test results confirm that the ABTA system adopting our method achieves more reliable performance than the original ABTA system when the training corpus is small.

Chapter 4

Sentence Identification of Biological Interactions using Generated Patterns and Optimized Parameters

In this chapter, we propose an approach to automatically extract sentence patterns containing relevant interaction involving concepts of molecular biology. This extraction is based on the assumption that biological interactions are articulated by a limited number of POS patterns embedded in sentences where entities/concepts are co-occurring. The extracted patterns are then applied to identify interaction sentences which describe interactions between biological entities. Our work aims to identify precise sentences rather than passages. Because of the nature of the patterns, we hope that some of the contextual information present in interaction sentences also play a role in the classification task.

The chapter is organized as follows: In Section 4.1, we introduce the problem of sentence identification of biological interactions. Section 4.2 describes an experimental system designed for this work. Sections 4.3, 4.4 and 4.5 elaborate the approaches and algorithms applied in the system. The performance of the system is evaluated in Section 4.6. Finally, Section 4.7 summarizes this work and introduces future directions.

4.1 Problem Description

An important task in information extraction (IE) in biological science is to identify sentences from research communications that contain important interactions between biological entities. The type of interaction of interest includes protein-protein, protein-DNA, gene regulations and other interactions between macromolecules. This work broadens the definition of the term "interaction" to include other types of concepts that are semantically related to cellular components and processes. This contrasts with the past efforts focusing strictly on molecular interactions [49, 71]. We hypothesize that identifying the relationships between concepts of molecular biology

will facilitate the building of knowledge models, improve the sensitivity of IE tasks and ultimately facilitate the formulation of new hypotheses by experimentalists.

The extraction of interactions is based on the heuristic premise that interacting concepts co-occur within a given section of text. The challenge is that co-occurrence certainly does not guarantee that a passage contains an interaction [5, 17, 18]. Co-occurrence is highly dependent on the definition of the section of text within which the target terms are expected to be found. A thorough comparison on the prediction of protein-protein interaction between abstract-level co-occurrence and sentence-level co-occurrence was undertaken [19]. It is demonstrated that abstract co-occurrence is more sensitive but less specific for interactions. At the cost of wide coverage, sentence co-occurrence increases the accuracy of interaction prediction. Since the ultimate goal of IE is to extract knowledge and accuracy is the most important aspect in evaluating the performance of such systems, it makes sense to focus the effort in seeking interaction sentences rather than passages or abstracts. Not every co-occurrence in sentences implies a relationship that expresses a fact. In the 2005 Genomics Track dataset, 50% of all sentence co-occurrences of entities correspond to definite relationships while the rest of the co-occurrences only convey some possible relationships or contain no relationship of interest [18]. Therefore, more sophisticated text mining strategies are required to classify sentences that describe interactions between co-occurring concepts.

In the BioCreative II challenge [72], participating teams were asked to determine whether a given passage of biological texts contained information about the interaction between two proteins. Although the best F-score of 78% was reported, this classification task worked at the abstract level and the interacting protein pairs were not required to be extracted. The task for the Learning Language in Logic (LLL'05) challenge [73] was to build systems that extract interactions between genes or proteins from the biological literature. From individual sentences annotated with agent-target relations and other linguistic information, patterns or models had to be learned to extract these interactions. The task focused on extracting only the interacting partners. The context of an interaction may also be critical to the validity of the extracted knowledge since not all statements found in the literature are facts.

4.2 Overview of System Design

In this work, we have designed an experimental system to extract biological interaction patterns and identify sentences that contain these interactions. Figure 4.1 illustrates the overall data flow of the system, which consists of three major modules: biological text preprocessing, interaction pattern extraction, and interaction sentence identification.

A text preprocessing reformats the original texts into a set of candidate sentences. A pattern learning method is then proposed to automatically extract the representative patterns of biological interactions. The obtained patterns are then used to identify sentences that describe biological interactions. Poor performance during preprocessing will have detrimental effects on later stages. In the following sections, we will describe each component of the system in detail.

4.3 Biological Text Preprocessing

Processing text from the biological literature is more challenging than from the general English texts. This is due to the domain-specific terminology and the ambiguity in some technical terms. Generally, several preprocessing steps need to be completed before performing text mining strategies. For our work, these include sentence preparation, Part-of-Speech tagging, biological entity recognition, and text chunking.

4.3.1 Sentence Preparation

Since our system aims to extract biological interaction patterns at the sentence level, a rule-based heuristic method is implemented to detect sentence boundaries [74]. Captions and headings that are not grammatically valid sentences are eliminated.

4.3.2 Part-of-Speech Tagging

POS tagging is then performed to associate each word in a sentence with its most likely Part-of-Speech tag. Because subsequent processing steps typically depend on the tagger's output, high accuracy at this level is crucial for success in later stages. A statistical tagger Lingua::EN::Tagger [70] is used to perform this task.

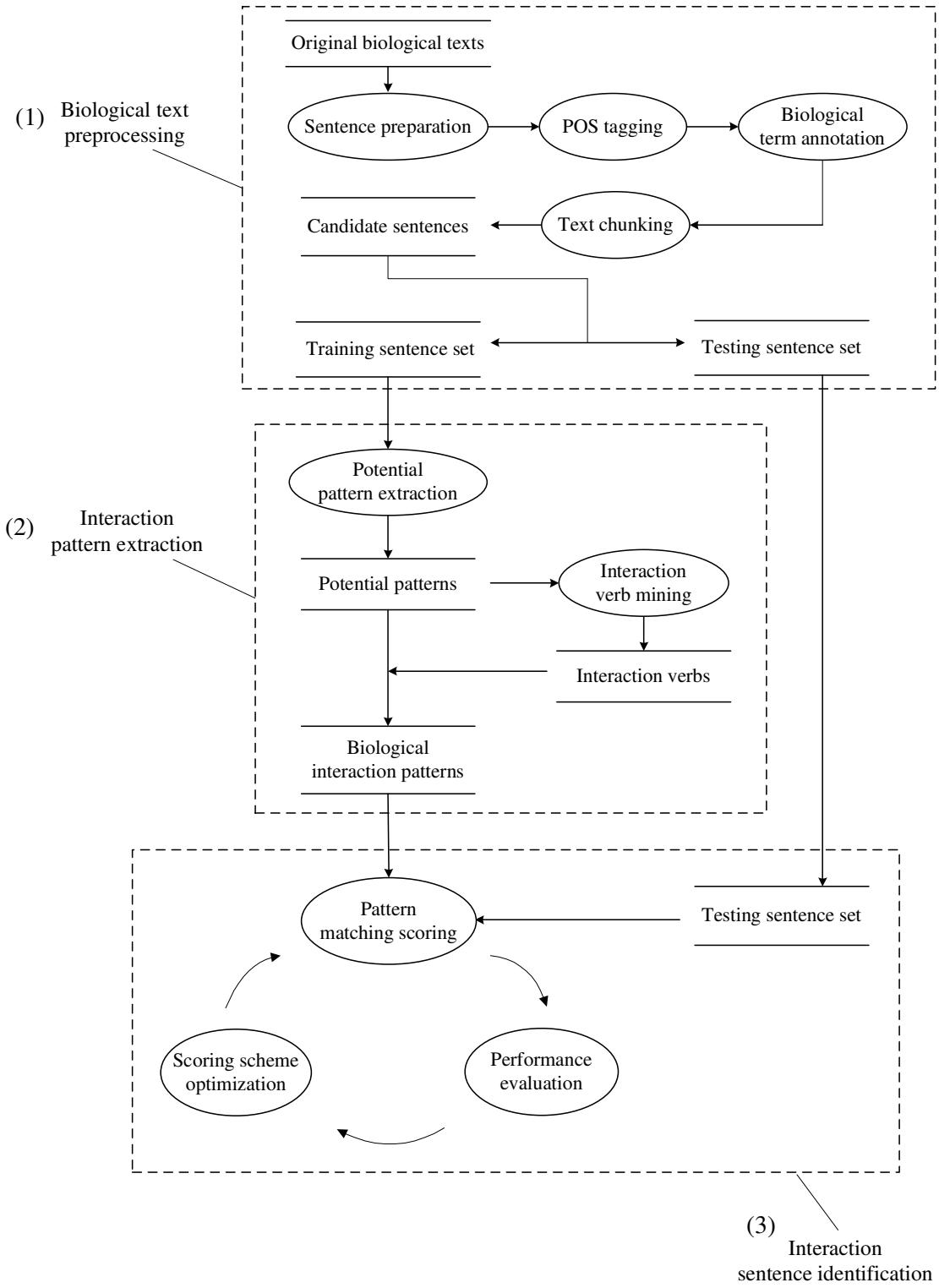


Figure 4.1: System Dataflow Diagram

4.3.3 Biological Entity Recognition

A learning-based biological entity recognition system, ABTA [16, 75], is embedded in our system. Given unstructured texts from the biological literature, ABTA locates biological entities based on five word position classes, "Start", "Middle", "End", "Single" and "Non-relevant". Therefore, multi-word entities should be in a consistent sequence of classes "Start (Middle)* End" while single word entities will be indicated by the class "Single". The type of entities includes molecules, such as genes, proteins and cell lines, and also biological processes. Examples of biological processes as entities are: "T cell activation" and "IL-2 gene transcription". We consider that a broader definition of biological entity will include more facts from literature, thus leading to more general use of interaction patterns for IE tasks.

ABTA considers the longest expression and ignores embedded entities. For example, "IL-2 receptor" is annotated for one kind of protein molecule and the embedded entity "IL-2" for another kind is not annotated. Also, parenthesized abbreviations of biological entities are not processed since their immediately preceding full forms will contribute to the patterns. We simply remove the parenthesized texts that follow the biological entities. Further, instead of distinguishing entities from their relevant biology concepts, a unified tag "*BIO*" is assigned to all the identified biological entities. We aim to discover patterns of the general interactions between biological concepts, not only the interactions between molecules, e.g., protein-protein interaction.

Tags like *NN*(noun) and *VB*(verb) are typically used to define entities and the action type of interactions, and thus they are indispensable. However, tags such as *JJ*(adjective), *RB*(adverb) and their comparative or superlative forms could occur at different positions in a sentence. We decided to remove these tags to prevent the combinatorial effect that these tags would induce within the set of extracted patterns. Meanwhile, other tags such as *PP*(sentence ending punctuation) were not considered in pattern definitions.

4.3.4 Text Chunking

Next, a rule-based text chunker [76] is applied on the tagged sentences to further identify base noun phrases *NP*, which are defined as nonrecursive noun phrases that

end after their nominal head and exclude any type of postmodification (e.g., prepositional phrases, genitives), such as <*NP*>oxygen production</*NP*> and <*NP*>long enhancer transcripts</*NP*>, and combine verbal elements into their verbal unit *VB*, for instance <*VB*>have been identified<*VB*>. This allows us to focus on the holistic structure of each sentence. Text chunking is not applied on the identified biological entities although they might also be labeled as *NPs* since most biological entities are linguistically expressed within noun phrases. In order to achieve more generalized interaction patterns, a unified tag "*VB*" is used to represent every verbal unit instead of employing different tags for various tenses of verbs.

As a result of preprocessing, every sentence is represented by its generalized form as a sequence of corresponding tags consisting of POS tags and predefined tags. Table 4.1 summarizes the main tags in the system.

Tag name	Tag description	Tag type
BIO	Unified tag for biological entities	Predefined
NP	Base noun phrase	Predefined
VB	Verbal unit	Predefined
IN	Preposition, subordinating conjunction	POS
CC	Coordinating conjunction	POS
TO	to	POS
PPC	Punctuation comma	POS
PRP	Determiner of possessive second	POS
DET	Determiner	POS
POS	Possessive	POS

Table 4.1: Main tags used in the system

For a sentence extracted from MEDLINE [2] "IL-2 gene expression and NF-kappa B activation through CD28 requires reactive oxygen production by 5-lipoxygenase." (MEDLINE: 95369245), the preprocessed result is:
<*BIO*>IL-2 gene expression</*BIO*> <*CC*>and</*CC*> <*BIO*>NF-kappa B activation </*BIO*> <*IN*>through</*IN*> <*BIO*>CD28</*BIO*> <*VB*>requires</*VB*>
<*NP*>oxygen production</*NP*> <*IN*>by</*IN*> <*BIO*>5-lipoxygenase</*BIO*>

A biological interaction tends to involve at least three objects: a pair of co-occurring biological entities connected by a verb which specifies the action type of the interaction. Therefore, a constraint is applied that only sentences satisfying form

"BioEntity A – Verb – BioEntity B" will be preserved as candidate sentences to be further processed in the system. It is possible that the presence of two entities in different sentence structures implies a relationship. However, this work assumes the underlying co-occurrence of two concepts and a verb in the interest of improving the classification accuracy. This type of "object-relation-object" inference is also assumed in other work [49, 61, 77, 78].

The obtained candidate sentences are split into training and testing sets. The training set is used to extract the representative patterns of biological interactions. The testing set is prepared for identifying sentences that evidently describe biological interactions.

4.4 Interaction Pattern Extraction

4.4.1 PATRICIA Trees

The method we propose to extract interaction patterns from candidate sentences is based on the use of PATRICIA trees [65]. In our work, a PATRICIA tree is used for the first time to facilitate the automatic extraction of interaction patterns. All the training sentences are inserted and stored in a generic PATRICIA tree. From this tree, the common patterns of POS tags can be efficiently stored and the tree structure used to compute relevant usage statistics.

4.4.2 Potential Pattern Extraction

The premise of this work is that there is a set of frequently occurring interaction patterns that match a majority of stated facts about molecular biology. In this work, a *biological interaction pattern* is defined as follows:

Definition 4.1. (Biological Interaction Pattern) A biological interaction pattern *bip* is a sequence of tags defined in Table 4.1 that captures an aggregate view of the description of certain types of biological interactions based on the consistently repeated occurrences of this sequence of tags in different interaction sentences. $BIP = \{bip_1, bip_2, \cdots, bip_k\}$ represents the set of biological interaction patterns.

We first extract potential interaction patterns by populating a PATRICIA tree using training sentences. Every node in the tree contains one or more system tags, which is the preceding tag sequence of its descendant nodes in each sentence. Every sentence is composed of a path of system tags from the root to a leaf. Hence, we propose that the sequence of system tags that can be formed from traversing the nodes of the tree is a potential pattern of biological interactions.

Figure 4.2 illustrates a simple example of the process of extracting potential interaction patterns from a PATRICIA tree populated by the tagged forms of four simplified sentences discussing various interactions between different biological entities symbolized as A, B, C, and D. Dotted circles represent leaf nodes while solid circles denote internal nodes. By traversing the nodes of the tree, 6 potential patterns are recursively retrieved: (1) *BIO CC BIO VB BIO* (2) *BIO VB* (3) *BIO VB BIO* (4) *BIO VB IN BIO* (5) *BIO VB BIO IN BIO* (6) *BIO VB BIO CC VB BIO*. Meanwhile, the occurrence frequency of each pattern is also retrieved and recorded from the tree.

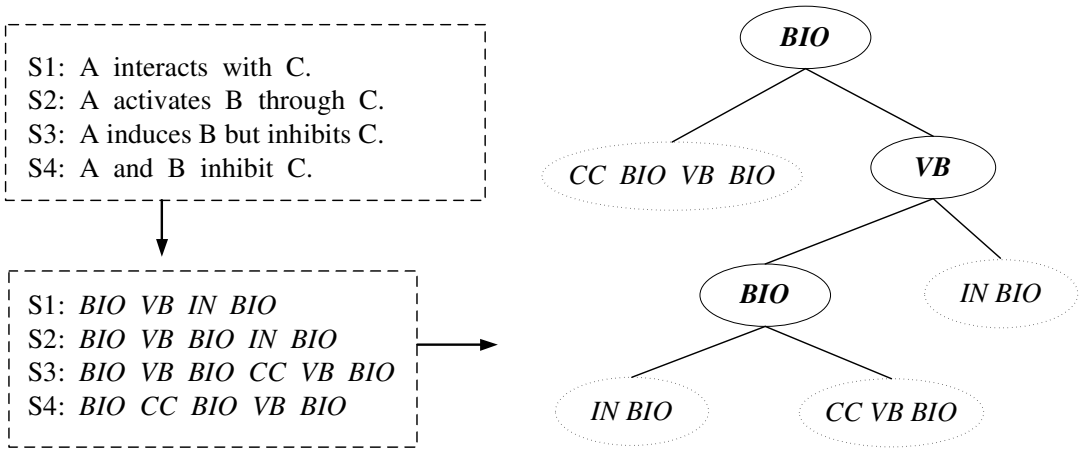


Figure 4.2: Example of Potential Pattern Extraction

A frequency threshold f_{min} is used as a constraint to filter out patterns that occur less than f_{min} times. It has been demonstrated that if an interaction is well recognized, it will be consistently repeated [49, 71]. While maintaining the size of the pattern set, the generalization and the usability of patterns can be also controlled by tuning f_{min}. The larger the threshold is, the more accurate patterns are. Further, some filtering rules are adapted to control the form of a pattern and enhance the quality of the discovered patterns, such as if a pattern ends with a tag *IN*, *VB*, *CC* or

TO, the pattern will be rejected. Flexibility in setting this threshold can be applied to meet special demands. As a result, the only valid pattern in the above example is "*BIO VB BIO*", if we set $f_{min} = 2$. For our system, we found that about 70% of patterns extracted from the training set appeared at least 5 times. We consider that these patterns are representative patterns, which have been intensively employed in expressing biological interactions. Therefore, we finally set $f_{min} = 5$. Algorithm 2 shows our pattern learning method which has a time complexity of $O(n)$ in the size of candidate sentences, n.

Algorithm 2 Patricia-Tree-based Extraction of Biological Interaction Patterns

Input: Candidate Sentences $CS \in$ Biological text; a frequency threshold f_{min}; a set of filtering rules FR

Output: BIP : Set of biological interaction patterns

```
1:  PT ← ∅                                              //PT : Patricia Tree
2:  BIP ← ∅
3:  for all sentences s ∈ CS do
4:      PT ← Insert(s)                                  //Populating Patricia Tree
5:  for all nodes n_i ∈ PT do
6:      bip_i ← Pattern(n_i) //Concatenating tags in nodes from root to n_i, which is a
7:      //potential pattern
8:      if Count(bip_i) ≥ f_min and bip_i does not meet FR then
9:          //Count(bip_i) returns the number of occurrences of bip_i;
10:         BIP ← bip_i
```

4.4.3 Interaction Verb Mining

Although the obtained patterns are derived from the candidate sentences possessing the form "BioEntity A – Verb – BioEntity B", some of them may not contain facts about biological interactions. This is possible if the action verbs do not describe an interaction. Quite a few verbs, for instance, "report", "believe", and "discover", do not define interactions at all, but only serve a narrative discourse purpose. Therefore, mining the correct interaction verbs becomes an important step in the automatic discovery of patterns.

An automatic approach was proposed in [78] to discover interaction verbs that code gene and protein interactions in molecular biology articles. The authors applied statistical tests and a logistic regression statistical model to determine whether a given verb was an interaction verb. The features used in the experiments included the frequencies of a verb before and after gene or protein names, and the frequencies of the verb in different domains. Another simpler method was applied to mine interaction verbs by raising the threshold value of the frequency of the extracted interaction patterns [36]. The authors assumed that a verb in a pattern is more likely to act as an interaction verb when the frequency of the pattern is comparatively high. The mined verbs were then manually evaluated by domain experts to remove inaccurate candidates.

We perform the latter method to mine a list of interaction verbs. This will be used to further improve the relevance of achieved patterns by filtering out patterns formed by the sentences in which the action verbs are not on the list.

4.5 Interaction Sentence Identification

Identifying interaction sentences is treated as a classification problem to differentiate between interaction sentences and non-interaction sentences. Each sentence in the training set is identified as either containing an interaction or not. An interaction is a binary relationship between two concepts relevant to molecular or cell biology.

4.5.1 Pattern Matching Scoring

We first perform pattern matching by conversion of each sentence to be classified into a sequence of tags. This sequence is then compared to every previously extracted pattern. This is done using sequence alignment which calculates the degree of the similarity of a sentence to an interaction pattern. Our hypothesis is that the alignment scores can be used to discriminate interaction sentences from other types of sentences.

The language used in this discussion borrows heavily on the language used in the pairwise sequence alignment literature. The scoring scheme involved in the process of pattern matching consists of penalties for introducing gaps, match rewards and mismatch penalties for a selection of system tag pairs. Table 4.2 presents a practical

scoring scheme derived from previous experiments for main tags used in the system. Penalties and rewards are denoted respectively by negative and positive values.

Tag	Gap	Match	Mismatch
BIO	-10	+8	-3
NP	-8	+6	-3
VB	-7	+7	-3
IN	-6	+5	-1
CC	-6	+5	-1
TO	-1	+5	-1
PPC	-1	+3	-1
PRP	-1	+3	-1
DET	-1	+3	-1
POS	-1	+3	-1

Table 4.2: An alignment scoring scheme for system tags

As a variation of global alignment [79], an end-space free alignment algorithm [80] is implemented to facilitate the comparison of patterns and testing sentences. It assumes that gaps at the beginning or ending positions of a sentence always have a cost of zero. This enables the detection of embedded interactions within longer sentences. In this work, the shortest pattern is selected when multiple patterns align with the same score. As a result, each sentence is assigned to its most compact pattern that maximizes the alignment score. Therefore, high-scoring interaction sentences should be distinguishable from non-interaction sentences which are scoring low against all considered patterns. Essentially, this procedure can be seen as a variation of the k Nearest Neighbors (kNN) classification method, with $k = 1$.

4.5.2 Performance Evaluation

We then evaluate whether the alignment scores can be used to classify the testing sentences. We have proposed two independent evaluation measures: statistical analysis (SA) and classification accuracy (AC).

Since it is assumed that interaction sentences achieve much higher alignment scores in pattern matching than non-interaction sentences, SA measures whether the scoring difference between the mean of interaction sentences and the mean of non-interaction sentences should be attributed to chance, or whether it is statistically significant. If

the scoring difference between the means is statistically significant, there will be a tendency that interaction sentences outscore non-interaction sentences in alignment. Hence, it would be reliable to use alignment scores to classify testing sentences.

Although non-interaction sentences come from the same documents as interaction sentences, we assume that interaction sentences and non-interaction sentences are two independent samples. Meanwhile, the Shapiro-Wilk test [81] package in SAS (v8.02) [82] was employed to test the normality of the score distribution for the two samples. Both distributions are normal distribution. Statistical analysis is then performed based on the following theory. If $\overline{x}_1$ and $\overline{x}_2$ are the means of the two samples, the mean and the standard deviation of the sampling distribution of the statistic $\overline{x}_1 - \overline{x}_2$ are:

$$\mu_{\overline{x}_1-\overline{x}_2} = \mu_1 - \mu_2 \quad and \quad \sigma_{\overline{x}_1-\overline{x}_2} = \sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$$

where μ_1, μ_2, σ_1 and σ_2 are the means and the standard deviations of the two samples respectively.

Then, the formula for statistical two-sample z test is given in the Eq.(4.1) with the null hypothesis $\mu_1 - \mu_2 = \delta$.

$$z = \frac{\overline{x}_1 - \overline{x}_2 - \delta}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \tag{4.1}$$

Although δ can be any constant, in this work we test only the null hypothesis that there is no scoring difference between the means of interaction and non-interaction sentences, namely $\delta = 0$. The conventional value of 0.05 is used for the level of significance α in the analysis.

A comparatively large z will lead to the rejection of the null hypothesis. In other words, the scoring difference is statistically significant. Consequently, interaction sentences can be separated from non-interaction sentences according to alignment scores. In practice, it is reasonable to assume that the two classes are not perfectly separable but that the scoring scheme can be optimized to minimize the overlap of the distribution of scores for both classes of sentences. Under the assumption of normality, maximizing z would accomplish this task.

Conversely, AC measures the proportion of correctly classified testing sentences,

including both interaction and non-interaction sentences, to the total testing sentences. An appropriate threshold T is determined for obtained alignment scores to differentiate between interaction and non-interaction sentences, and to facilitate the calculation of classification accuracy.

While two evaluation measures are proposed, it is not possible to evaluate the performance without correctly pre-labeled testing sentences. We decided to manually classify the testing sentences in advance by assigning each sentence an appropriate label of interaction or non-interaction. This work was done by me and Dr. Christian Blouin who has a Ph.D. degree in molecular biology, and an agreement was reached before pre-labeling the sentences, that is, the class label assigned to each testing sentence must be in accordance with the main intention of the sentence. We studied the contents of each sentence, judged the principal idea, and then concluded the class label.

4.5.3 Scoring Scheme Optimization

NP-hardness

The scoring scheme applied in pattern matching has a crucial impact on the performance of interaction sentence identification since it is based on the pattern matching scores directly determined by the different costs specified in the scoring scheme. Optimizing the scoring scheme to minimize the overlap of the distribution of scores is not a trivial problem. So far, an empirical or arbitrary scoring scheme was adopted in previous research publications for the pairwise alignments [36,58]. We have proved that finding the optimal solution for a variation of this problem is NP-hard by reducing a well-known NP-hard problem 3-SAT to the problem [52]. Therefore, our hypothesis is that this problem is NP-hard as well. This is an important claim to justify the use of heuristic methods for determining the best parameter settings.

Heuristic approach using genetic algorithm

A genetic algorithm (GA) [83] is used as a heuristic method to optimize parameters of the scoring scheme for sentence classification. The Perl package AI::Genetic [84] is applied to perform the canonical genetic algorithm.

In practice, our GA works with a population of potential solutions (scoring matrices). Ultimately, the resulting optimized scoring scheme is the solution scheme for sentence classification. The costs of penalties and rewards for different system tags are encoded by integer values within two predefined ranges: [-50, 0) and (0, 50], and assembled as a potential solution of scoring scheme, which consists of 30 parameters covering the costs for tags in the alignment as listed in Table 4.2. For each potential solution, the fitness function of GA performs the pattern matching procedure, evaluates the resulting alignment scores using the two evaluation measures SA and AC respectively, and returns a z value in terms of the formula (1) or a corresponding classification accuracy. GA iterates the fitness function with a goal of maximizing z value or classification accuracy. A predefined strategy of AI::Genetic, "rouletteSinglePoint", is specified to implement roulette-wheel selection and single-point crossover. The crossover and mutation probabilities are set to the empirical values, 0.95 and 0.01, according to the genetic algorithm package.

Our GA is set up to evolve for 100 generations, each of which consists of a population of 100 potential solutions of scoring scheme. GA starts with a randomly generated population of 100 potential solutions and proceeds until 100 generations are reached. The number of generations and the population size are decided with consideration of the runtime cost of evaluating the fitness function, which requires running the scoring algorithm with each sentence. GA evaluates all individuals in a population, and the population is replaced on a generational basis. The members of the population reproduce, and therefore their offspring must then be evaluated. Consequently, a large number of generations or a large population size would incur an expensive runtime cost of evaluation.

4.6 Results and Evaluation

This section starts with the description of our experimental dataset. Biological text preprocessing results are then presented in section 4.6.2 followed by results of interaction pattern extraction in which representative interaction patterns are derived from refining the extracted potential patterns based on two types of annotations. Next, results of interaction sentence identification are described in detail. In the end, we conduct experiments to evaluate another state-of-the-art pattern generating

algorithm in order to compare with our proposed method for interaction sentence identification.

4.6.1 Dataset

Our experiments have been conducted on the GENIA corpus (v3.02) [21], the largest annotated corpus in the molecular biology domain available to the public. It consists of 2,000 biological research paper abstracts selected from the search results in the MEDLINE database. It is intended to cover biological reactions concerning transcription factors in human blood cells. The carefully curated information of sentence segmentation, word tokenization, POS tagging and biological entity recognition has also been encoded in the corpus.

4.6.2 Biological Text Preprocessing Results

Evaluated using the inherently equipped annotation information, our system achieves nearly 99% accuracy on segmenting sentences and extracts 18,355 sentences from the GENIA corpus. About 99% of sentences in the corpus end with a period, and nearly 94% of periods are sentence boundaries (6% at the end of abbreviations and about 0.3% as both).

Furthermore, the system obtains an overall POS tagging accuracy of 91.0% on 364,208 individual words. Table 4.3 shows the tagging results of main POS tags. We noticed that the tagging information encoded in the GENIA corpus is not always consistent throughout the whole corpus, thus introducing detrimental effects on the tagging performance. For instance, in the GENIA corpus the word "stimulated" is tagged as *VBN* for terms "stimulated human endothelial cells" (MEDLINE:95202809) and "minimally stimulated T cells" (MEDLINE:92156807) but tagged as *JJ* for terms "stimulated IL-5 promoter activation" (MEDLINE:97407957) and "Fc gamma RIIA stimulated cells" (MEDLINE:96289501). Also, considering that the tagger used in the system is developed and parameterized according to the general English domain, porting this tagger to biology domain is accompanied by some loss in performance. We expect that retraining our tagger on annotated biological language corpora will improve the tagging performance.

Tag name	Tagging accuracy (%)	Total count
NN	97.5	134,044
IN	99.8	62,348
JJ	70.5	46,414
NNS	99.6	32,522
CC	92.2	17,449
VBN	86.8	14,702
RB	88.5	12,393
VBD	96.0	10,342
TO	100.0	7,616
VB	93.9	5,452
VBG	87.6	4,940
VBZ	89.0	9,172
VBP	89.9	6,814
Overall	91.0	364,208

Table 4.3: Part-of-Speech tagging accuracy

The system reaches an F-score of 0.705 on annotating all biological entities including both multi-word and single word entities. After performing text chunking, the system produces a set of candidate sentences. We further perform text chunking on the GENIA corpus based on its encoded annotations and use the resulting set of sentences for the subsequent experiments to provide a gold standard to which results produced based on system annotations can be compared. Table 4.4 presents some statistics of the preprocessed dataset. For GENIA annotations, we extracted 16,272 candidate sentences which possess the form "BioEntity A – Verb – BioEntity B". For system annotations, we obtained 1,250 more candidate sentences than GENIA annotations due to the annotation differences. For each type of annotation, we randomized the candidate sentence set and chose 12,525 candidate sentences as the training set to extract biological interaction patterns. The rest of candidate sentences are prepared as the testing set for interaction sentence identification.

4.6.3 Interaction Pattern Extraction Results

For our system, a frequency threshold $f_{min} = 5$ is used to filter out the potential patterns that appear less than 5 times in the training set. Evaluated by domain experts, a list of 300 interaction verbs and a list of 700 non-interaction verbs are obtained from

Attributes	GENIA	System
Total preprocessed sentences	18,545	18,355
Candidate sentences	16,272	17,525
Training set sentences	12,525	12,525
Testing set sentences	6,020	5,000

Table 4.4: Statistics of experimental dataset

12,525 training sentences with GENIA annotations. Further, inflectional variants of the verbs are also added into the lists. For example, for an interaction verb "stimulate", its inflectional variants including "stimulates", "stimulated" and "stimulating" are added. Table 4.5 gives some examples of mined interaction verbs.

activate	consolidate	inhibit	moderate	repress
antagonize	depress	intensify	modulate	reproduce
augment	degenerate	interact	neutralize	translocate
bind	dissociate	interfere	prevent	trigger
block	hamper	localize	prohibit	ubiquitinate
catalyze	induce	magnify	reduce	unphosphorylate

Table 4.5: Examples of mined interaction verbs

Refined by the filtering rules and the list of interaction verbs, a final set of representative patterns of biological interactions are obtained. We performed our proposed pattern learning method on training sentences of both GENIA and system annotations. As shown in Table 4.6, while system annotations achieve more patterns than GENIA annotations, there are 97 common patterns between them.

Attributes	GENIA	System
Potential patterns ($f_{min} = 5$)	241	329
Extracted patterns ($f_{min} = 5$)	209	302

Table 4.6: Pattern extraction results

Moreover, Table 4.7 lists the 10 most frequent interaction patterns based on GENIA annotations. For instance, a training sentence conforming to the second pattern is "The expression of the QR gene is regulated by the transcription factor AP-1." (MEDLINE: 96146856).

Pattern count	Pattern form
264	*BIO VB BIO IN BIO*
261	*NP IN BIO VB IN BIO*
182	*NP IN BIO VB BIO*
162	*BIO IN BIO VB IN BIO*
160	*BIO VB IN BIO IN BIO*
143	*NP IN BIO VB IN NP IN BIO*
142	*NP VB IN BIO VB BIO*
138	*PRP VB IN BIO VB BIO*
126	*BIO VB NP IN BIO IN BIO*
121	*NP IN BIO VB NP IN BIO*

Table 4.7: Extracted biological interaction patterns

4.6.4 Interaction Sentence Identification Results

Four hundred sentences were randomly extracted from the testing set. All of these sentences contained the "BioEntity A – Verb – BioEntity B" set of system tags. Each was manually labeled into two classes: interaction and non-interaction. The distribution of class labels of the sample sentences is shown in Table 4.8.

Class label	400 sentences	
	Number	Proportion(%)
Interaction	211	52.75
Non-interaction	189	47.25

Table 4.8: Class distribution of sample sentences

Validation of convergence property of GA

We generated an optimized scoring scheme using a GA (100 individuals, 100 generations) on the 400 testing sentences and obtained its classification. To simulate a perfect classifier, we treated this classification as absolute truth in a second GA optimization of the scoring scheme on the testing sentences. The resulting scheme converged with a 97.75% classification accuracy. This provides an evidence of the stability of the problem with respect to the parameter values for the population size and the number of generations.

Comparison between two evaluation measures

We applied the evaluation measures, SA and AC, respectively in the fitness function of GA to the 400 testing sentences, and recorded the scoring scheme of every generation resulted from GA. Then, the classification accuracy in terms of each scoring scheme derived from SA is computed for comparison. Figure 4.3 presents the distribution of achieved classification accuracy in terms of each scoring scheme optimized by GA. This comparison is done with respect to the generation and evaluated on 400 testing sentences using the annotation from the GENIA corpus.

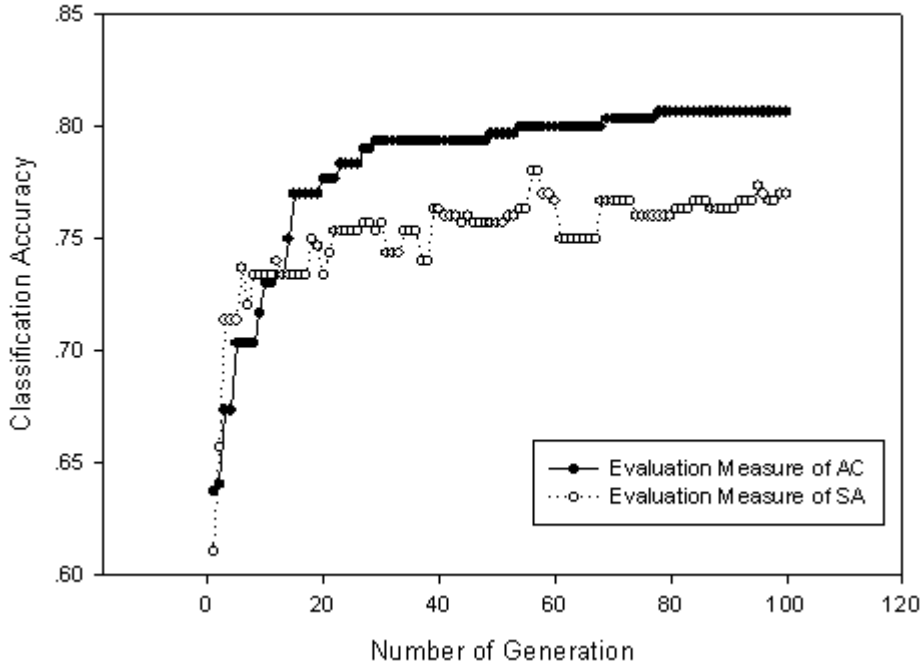


Figure 4.3: Classification Accuracy Comparison between Two Measures

The achieved classification accuracy for AC increases in a ladder-like fashion, and generally outperforms the classification accuracy derived by SA. It reaches its highest classification accuracy 80.75% from the 78th generation. In contrast, for SA the increase of z does not correspond to the increase of classification accuracy. The highest classification accuracy, 78%, is reached at the 56th and 57th generations. Therefore, AC is considered more efficient with the system and becomes our final choice of the evaluation measure used in the fitness function of GA.

Results of sentence identification

Since our sample set is comparatively small, we implemented a strategy of cross-validation, and calculated the average performance over 8 runs of labeled sentences. First we divided the 400 sentences into 8 equal subsets. Then we used 7 subsets, 350 sentences, as training data, and the remaining 50 sentences as testing data. Based on the obtained interaction patterns, GA results in an optimized scoring scheme on the 350 sentences along with its associated scoring threshold T, which are then applied together to the other 50 testing sentences for sentence classification. This procedure was repeated 8 times. Classification performances were averaged over these 8 runs to produce a single estimation. Table 4.9 and Table 4.10 report the averaged system performances and the corresponding standard deviation (STDEV) on the sample set respectively to both GENIA and our system annotations.

Experimental Results	GENIA annotation			
	Interaction	STDEV	Non-interaction	STDEV
Precision	0.736	0.052	0.898	0.080
Recall	0.921	0.075	0.694	0.066
F-score	0.816	0.042	0.781	0.070
Overall AC(%)	80.25	0.046		

Table 4.9: Cross-validated performance based on GENIA annotation

Experimental Results	Our system annotation			
	Interaction	STDEV	Non-interaction	STDEV
Precision	0.711	0.056	0.750	0.062
Recall	0.800	0.068	0.626	0.035
F-score	0.747	0.054	0.670	0.047
Overall AC(%)	72.50	0.043		

Table 4.10: Cross-validated performance based on our system annotation

Table 4.9 shows that when using the GENIA annotations the system achieves on average 0.816 F-score in identifying interaction sentences and an overall AC of 80.25%, which is much higher than the proportion of either interaction or non-interaction sentences in the 400 sentence subset. This indicates that the system performs well on both classes. Meanwhile, in 100 generations GA is not able to evolve a scoring scheme that leads to an AC above 86%. We have also tested 20 more generations but AC

shows no improvement when GA reaches the 120th generation. This indicates that the scoring scheme and threshold resulted from the 350 training sentences cannot provide enough distinction to correctly classify some of the 50 testing sentences. Compared to GENIA annotations, our system annotations achieve a lower performance shown in Table 4.10. We attribute the difference to the accuracy loss of system annotations in the preprocessing steps as inaccurate annotations will lead to inappropriate patterns, thus harming the performance of sentence identification.

There are a number of preprocessing steps that affect the final classification performance. However, even assuming an ideal preprocessing of the unstructured text, our method relies on the assumption that all interaction sentences are articulated by a set of POS patterns that are distinct to all other types of sentences. The manual annotation of the training/testing set was a difficult task, so it is reasonable to assume that this will also be difficult for the classifier. The use of passive voice and the common use of comma splicing within patterns makes sentence-level classification an especially difficult task. Another source of interactions that our system cannot identify are implied and assume a deeper semantic understanding of the concepts themselves. Other sentences are long enough that the interaction itself is merely a secondary purpose to another idea. All of these factors pose interesting challenges for future development of this work.

Moreover, we also experimented with 10 empirical scoring schemes derived from previous experiments on the 400 sentences, including the scheme shown in Table 4.2. Several fixed thresholds were attempted for obtained alignment scores to differentiate between interaction and non-interaction sentences. Without using GA to optimize parameters of the scoring scheme, the best performance of 10 empirical schemes is an overall *AC* of 66%, which has been outperformed at the 3rd generation of GA optimization with GENIA annotations.

Impact of verb tags

Instead of using the unified tag "*VB*" for all verbal units, we further employed different tags to differentiate between interaction and non-interaction verbs, and investigated the impact of the tags on the system performance. "*IVB*", "*NONIVB*", and "*VB*" are used to represent interaction verbs, non-interaction verbs, and verbs that are not

on both lists.

Following the same experimental procedures, candidate sentences with new verb tags are prepared and corresponding interaction patterns are then derived. An additional filtering rule is applied if a pattern does not contain tag "*IVB*", the pattern will be rejected. Table 4.11 shows the pattern extraction results with new verb tags.

Attributes	GENIA	System
Potential patterns ($f_{min} = 5$)	180	289
Extracted patterns ($f_{min} = 5$)	116	154

Table 4.11: Pattern extraction results with new verb tags

New verb tags generally result in fewer interaction patterns than the unified tag. Based on fewer patterns, the system adopting new tags achieves a noticeable improvement in performance on testing sentences in terms of both GENIA and our system annotations compared to the original system. Table 4.12 and Table 4.13 report the detailed results.

Experimental	GENIA annotation			
Results	Interaction	STDEV	Non-interaction	STDEV
Precision	0.894	0.086	0.803	0.044
Recall	0.806	0.033	0.891	0.086
F-score	0.845	0.046	0.842	0.051
Overall $AC(\%)$	84.50	0.042		

Table 4.12: Cross-validated performance based on GENIA annotation with new tags

Experimental	Our system annotation			
Results	Interaction	STDEV	Non-interaction	STDEV
Precision	0.778	0.044	0.752	0.068
Recall	0.821	0.076	0.739	0.057
F-score	0.796	0.040	0.742	0.064
Overall $AC(\%)$	77.50	0.034		

Table 4.13: Cross-validated performance based on our system annotation with new tags

4.6.5 System Performance Comparison

Within the framework of our system, we further conducted experiments on the same dataset for sentence identification using interaction patterns generated by another pattern generating algorithm (PGA) [36] in order to compare with the performance of patterns obtained by our proposed pattern learning method.

The algorithm is based on sequence alignment, which calculates a consensus sequence in addition to the similarity score. This consensus sequence represents all common parts (POS tags and their positions) in the aligned sequences and could be used directly to form a pattern. The local alignment algorithm [51] was implemented for the alignment and scoring of sentence pairs [36]. Thus, the aligned sentences could be quite dissimilar overall, but contain regions that are highly similar.

In our implementation, PGA iterates over all pairs of candidate sentences in the training set and calculates the best alignment for each pair in terms of the cost scheme of gap penalties proposed [36]. Each consensus sequence from the optimal alignment of each pair forms a pattern. The number of occurrences of each pattern is also calculated as the support for each. In principle, the maximum number of patterns for n sentences is $n(n-1)/2$, but in practice not all will be generated since a set of different alignments can lead to the same consensus sequence. On the other hand, if multiple optimal alignments exist when aligning a sentence pair, all alignments will be considered to form patterns. The filter rules proposed in [36] are also applied in our implementation. All patterns below a minimum support are removed from the set of generated patterns.

Algorithm 3 shows the procedure of the pattern generating algorithm, which has a time complexity of $O(n^2)$ in the size of candidate sentences, n. Hence, our proposed pattern learning method is much more efficient when dealing with large collections of biological texts.

As shown in Table 4.14, PGA produces a large number of patterns, even with $f_{min} = 5$ and other filtering criteria. There are 37,319 common patterns between two types of annotations.

In order to make a direct comparison, we decided to experiment with the same number of interaction patterns. For GENIA annotations, we chose the most frequent 209 patterns generated by PGA to compare with the 209 patterns by our method. For

Algorithm 3 Pattern Generating Algorithm

Input: Candidate Sentences $CS \in$ Biological text; a frequency threshold f_{min}; a set of filtering rules FR

Output: BIP : Set of biological interaction patterns

```
1: BIP ← ∅
2: for all (s_i, s_j) ∈ CS(i ≠ j) do
3:     alignment_{i,j} ← Align(s_i, s_j)          //Alignment for s_i and s_j
4:     bip ← Consensus(alignment_{i,j})           //Consensus sequence to form a pattern
5:     if bip does not meet FR then
6:         if bip ∉ BIP then
7:             BIP ← bip
8:             count_bip = 1
9:         else
10:            count_bip ++
11: for all bip ∈ BIP do
12:    if count_bip < f_min then
13:        remove bip from BIP
```

Attributes	GENIA	System
Potential patterns ($f_{min} = 5$)	476,600	387,302
Extracted patterns ($f_{min} = 5$)	176,082	88,800

Table 4.14: Pattern extraction results of PGA

system annotations, two sets of 302 patterns are employed. Further, it is found that for GENIA annotations there are 96 common patterns between the two sets of 209 patterns, and there are 153 common patterns between the two sets of 302 patterns for our system annotations. 8-fold cross-validation is performed on the 400 testing sentences for sentence identification. Table 4.15 and Table 4.16 present the averaged results of sentence identification of PGA over 8 runs with respect to both GENIA and our system annotations.

The results show that patterns generated by PGA do not perform as well as patterns obtained by our method. We further experimented with new verb tags. Although new tags help to improve the performances, they are generally inferior

Experimental Results	GENIA annotation			
	Interaction	STDEV	Non-interaction	STDEV
Precision	0.706	0.046	0.846	0.066
Recall	0.870	0.069	0.635	0.053
F-score	0.776	0.043	0.724	0.057
Overall $AC(\%)$	75.50	0.047		

Table 4.15: Cross-validated performance of PGA based on GENIA annotation

Experimental Results	Our system annotation			
	Interaction	STDEV	Non-interaction	STDEV
Precision	0.677	0.049	0.738	0.057
Recall	0.814	0.072	0.565	0.043
F-score	0.732	0.053	0.634	0.051
Overall $AC(\%)$	69.50	0.044		

Table 4.16: Cross-validated performance of PGA based on our system annotation

to the results using patterns obtained by our method. We therefore infer that our proposed method is more efficient in producing biological interaction patterns to identify interaction sentences.

4.7 Summary

In this chapter, a novel approach is presented to automatically extract the representative patterns of biological interactions, which are then used to detect sentences that describe biological interactions. We have conducted the experiments on our designed system based on the GENIA corpus. By means of a genetic algorithm to define the scoring scheme, the system achieves on average 0.845 F-score using GENIA annotations and 0.796 F-score using our system annotations by cross-evaluating 400 sentences in which at least two biological concepts co-occur.

We further investigated the impact of different verb tags on the system performance. Based on fewer interaction patterns, the system adopting new tags achieves a noticeable improvement in performance on testing sentences in terms of both GENIA and our system annotations compared to the original system. By comparing with another pattern generating algorithm, we infer that our proposed method is more efficient in producing interaction patterns to identify interaction sentences.

Chapter 5

Biological Event Extraction using Subgraph Matching

In this chapter, we extract complex biological events from the scientific literature in tackling the primary task of the BioNLP'09 shared task on event extraction [1]. We represent sentences as grammatical dependency graphs and extract biological event rules from training sentences as minimal dependency graphs. We then investigate whether subgraph matching of event rules into testing sentences can be used to extract new biological events from the literature.

The chapter is organized as follows: In Section 5.1, we describe the problems in the task of biological relation extraction and introduce the BioNLP'09 shared task on event extraction. Section 5.2 elaborates the approaches and algorithms proposed in our graph matching-based method. Section 5.3 discusses important implementation issues. The performance of our method is evaluated and analyzed in Section 5.4. Finally, Section 5.5 summarizes this work and introduces future directions.

5.1 Problem Description

Much research in information extraction in the biological domain has focused on extracting semantic relations between molecular biology concepts [4, 5, 34–38]. State-of-the-art protein annotation methods have achieved reasonable success with a performance of 88% F-score [39]. A task of interest is to automatically extract protein-protein interactions (PPI). To date, most of the biological knowledge about these interactions is only available in the unstructured texts from scientific articles [6,85]. Although various approaches were proposed for the PPI extraction, the best-performing system from the BioCreative II challenge [86] only achieved a 29% F-score in identifying protein pairs in a sentence that have a biologically relevant relationship. This suggests that the problem of biological relation extraction is difficult and far from solved.

Sentences in the biological literature tend to be long and complex, and often

have long-range dependencies. Therefore, co-occurrence based or surface pattern based shallow analysis on biological texts suffers from either low precision or low recall [4, 6, 87]. As a result, full parsing has been explored as the basis for relation extraction to perform intensive syntactical and semantical analysis [4–6,34,35,37,61]. In the BioNLP'09 shared task on biological event extraction [1], 20 out of the total 24 participating teams resorted to a full parsing strategy, including all top 10 performing teams. However, most previous work extracts relevant relations based on a limited set of manually designed rules that map interpreted syntactic structures into the semantic relations [4,5,34,35,37,61]. We propose an approach to automatically learn rules that characterize a wide range of biological relations and events from a syntactically and semantically annotated corpus. Our approach is also based on full parsing of biological texts.

More recently, the dependency representation obtained from full parsing, with its ability to reveal long-range dependencies, has shown an advantage in biological relation extraction over the traditional Penn Treebank-style phrase structure trees [64]. Biological relations are generally extracted from the dependency representation by two main approaches. In one approach, the dependency representation is traversed and paths that contain the relevant terms describing the relations predefined in the rules are extracted as candidate relations [4,61,87]. In the other, relations are learned from the dependency representation using supervised machine learning based on specialized feature representations or kernels, encoded with dependency paths from the representation [40, 46, 47].

Graphs provide a powerful primitive for modeling biological data such as pathways and protein interaction networks [88–90]. As a result, subgraph matching has been intensively explored in the biological community to find a query graph of pathways in a graph database of known biological processes by measuring the graph similarity. Since the dependency representation maps straightforwardly onto a directed graph [91], properties and operations of graphs can be naturally applied to the problem of biological relation extraction. Rather than the above two approaches, we propose a graph matching-based approach using subgraph matching to extract biological relations and events from the scientific literature for the purpose of tackling the primary task of the BioNLP'09 shared task on biological event extraction. The

relation extraction is performed by matching the dependency representation of automatically learned rules to the dependency representation of biological sentences. This process is treated as a subgraph matching problem, which corresponds to the search for a subgraph isomorphic to a rule graph within a sentence graph.

5.1.1 BioNLP'09 Shared Task on Event Extraction

The BioNLP'09 shared task [1] focused on the recognition of biological events that appear in the biological literature. The goal of the shared task was to competitively evaluate information extraction (IE) systems targeting complex biological events. A biological event describes a change on the state of one or more biological molecules. It involves multiple participants or arguments of varying numbers. These arguments are assigned semantic roles and event types are determined based on existing biological ontologies. A biological event can be recursively embedded as a sub-event that functions as an argument in other events, thus facilitating the construction of complex biological networks.

When a biological event is described in scientific literature, we can analyze it by recognizing an *event type*, the *event trigger* which signals the event in texts, one or more *event arguments*, and the *source text* (ST), where the event is described. The source text is composed of tokens. We define *tokens* as finite strings of characters from a finite alphabet. The *alphabet* is a finite set of symbols Σ, or a subset of it. In practice, elements of the alphabet are ASCII characters. Tokens come from W, the set of all finite strings of characters from Σ. The source text is a finite sequence of tokens, i.e., any member of W^*.

Events are normally characterized by verbs or nominalized verbs that indicate the state change of a biological molecule or interactions between various biological entities. In the sentence "Tumor necrosis factor induced slightly c-fos and had almost no effect on c-jun and AP1." (MEDLINE: 1314139), an event is specified by the trigger verb "induced", and the event arguments are defined as "Tumor necrosis factor" and "c-fos". Semantic roles are assigned to the two arguments. The trigger "induced" has the first argument "Tumor necrosis factor", whose semantic role is an agent or cause, and the second argument "c-fos", whose semantic role is a theme. The type of the event is Positive_regulation, which is determined based on Gene Ontology (GO) [20]

and the GENIA ontology [21].

Since our focus is on description of biological events in text, we define a *biological event* in a way consistent with the shared task, which is as follows:

Definition 5.1. (Biological Event) A biological event is a four-tuple $e = (Type, Trigger, Arguments, ST)$. $ST \in W^*$, called the *source text*, is a sequence of tokens that contains the event; $Type \in T_e$ is an event type from a finite set of event types T_e; *Trigger* is a substring of tokens from ST that signals the event; *Arguments* is a non-empty, finite set of pairs (l, a) where $l \in L$ is a label from a finite set of semantic role labels L, and a is a token from ST, or another biological event.

For the BioNLP'09 shared task, T_e consists of nine event types defined in Table 5.1, and $L = \{\text{Theme, Cause}\}$. A *gold event* denotes a biological event where all the information has been manually annotated by domain experts.

Similarly to the previous shared tasks, LLL [73] and BioCreative [72], which focused on biological information extraction to seek relations between biological molecules, the BioNLP'09 shared task also addressed biological IE, but took a definitive step further toward finer-grained information extraction. While LLL and BioCreative targeted a rather simple representation of relations of biological molecules, i.e., protein-protein interactions, the BioNLP'09 shared task was concerned with the detailed behavior of biological molecules and semantically rich biological events. It was the first competitive evaluation of its kind in the BioNLP field as the extraction of complex biological events became possible only recently with the introduction of corpora containing the necessary annotations: the GENIA event corpus [44] and the BioInfer corpus [45].

The shared task focused on extraction of biological events particularly on proteins or genes, which were not distinguished in the task. In order to concentrate efforts on the novel aspects of the event extraction, it was assumed that the protein recognition had been already performed, and the task was thus equipped with a given set of gold protein annotation. In order to facilitate evaluation on different aspects of the overall task, the task was further divided into three subtasks to address event extraction at different levels of specificity [1]:

Task 1. Core event detection: involves the detection of typed events and assignment of given proteins as their primary arguments.

Task 2. Event enrichment: involves the recognition of entities that serve as secondary arguments to further specify the events extracted by Task 1.

Task 3. Negation and Speculation detection: involves the detection of negations and speculation statements regarding events extracted by Task 1.

The primary task, Task 1, was to detect biological events such as protein binding and phosphorylation, given only the annotation of protein names. It was required to extract type, trigger, and primary arguments of each event. This task is an example of extraction of semantically typed, complex events for which the arguments can also be other events. Such embedding results in a nested structure that captures the underlying biological statements more accurately compared to the previously prevailing approach of detecting binary interactions between biological entities [72,73]. In this work, we focus on Task 1 and propose a graph matching-based method to cope with the problem.

5.1.2 Dataset

For our experiments, the same datasets from the BioNLP'09 shared task are used. The organizers provided three human-curated datasets for training and evaluating participating systems. A training set and a development set are provided together for the purpose of training. They are prepared based on the publicly available portion of the GENIA event corpus [44] with the gold protein annotation and the gold event annotation given. There are 800 biological research abstracts in the training set and 150 abstracts in the development set. Also, a testing set of 260 abstracts is prepared from a held-out part of the same corpus and provided without the gold event annotation. The objective is to recreate the gold event annotation of the testing set based on the information induced from the training data. Results on the testing set are required to be submitted to the website of the shared task [92] in order to receive official evaluations on the testing data.

Table 5.1 shows the nine event types considered in the shared task. The event types were selected on the basis of their frequency and importance in the GENIA event corpus. Since these types are all related to protein biology, they take proteins

as their theme. The first three types are related to protein metabolism, i.e., protein production and breakdown. Phosphorylation is a typical protein modification event, while Localization and Binding are two types of molecular events. Regulation, including the sub-types Positive and Negative regulation, represents regulatory events and causal relations.

As shown in Table 5.1, the themes of events are primary arguments that are critical in its identification. For regulation events, the entity or event stated as the cause of the regulation is also regarded as the primary argument. The first five event types require only one theme, and the task can be cast as relation extraction between an event trigger and a protein name. Binding events are more complicated in requiring the detection of an arbitrary number of themes. Regulation events always take a theme argument and, when expressed, also a cause argument. As a unique feature of the BioNLP task, regulation events may take another event, namely sub-event, as its theme or cause.

	Event type	Primary arguments
1	Gene_expression	Theme(Protein)
2	Transcription	Theme(Protein)
3	Protein_catabolism	Theme(Protein)
4	Phosphorylation	Theme(Protein)
5	Localization	Theme(Protein)
6	Binding	$(\text{Theme(Protein)})^{+}$
7	Regulation	Theme(Protein/Event), $(\text{Cause(Protein/Event)})^{?}$
8	Positive_regulation	Theme(Protein/Event), $(\text{Cause(Protein/Event)})^{?}$
9	Negative_regulation	Theme(Protein/Event), $(\text{Cause(Protein/Event)})^{?}$

Table 5.1: Event types and primary arguments

Figure 5.1 illustrates an example of the format of the training data. Each target text file contains two lines, one for the title and the other for the abstract. The sentence segmentation is not provided and events may involve proteins that appear in different sentences. In the annotation file, each annotation is specified on a separate line. There are two types of annotation: (1) protein and event trigger annotations whose IDs begin with "T", and (2) event annotations whose IDs begin with "E".

The target text file:

```
RFLAT-1: a new zinc finger transcription factor that
activates RANTES gene expression in T lymphocytes.
RANTES (Regulated upon Activation, Normal T cell
Expressed and Secreted) is a chemoattractant cytokine
(chemokine) important in the generation of inflammatory
infiltrate and human immunodeficiency virus entry into
immune cells. RANTES is expressed late (3-5 days) after
activation in T lymphocytes.
```

The annotation file:

```
T3 Protein 0 7 RFLAT-1
T4 Protein 63 69 RANTES
T5 Protein 104 110 RANTES
     .
     .
     .
T7 Gene_expression 75 85 expression
T8 Positive_regulation 53 62 activates
E1 Gene_expression:T7 Theme:T4
E2 Positive_regulation:T8 Theme:E1 Cause:T3
     .
     .
     .
```

Figure 5.1: Format of Training Data

Each annotation is assigned a unique ID. The protein and event trigger annotations are presented as a four-tuple of entity type, offset-begin, offset-end and text span. The offset-begin is the index of the first character in the entity. The offset-end is the index of the first character after the entity. The text span is then the text substring specified by the offset-begin and the offset-end. Each event annotation is expressed as an N-tuple of event-type and arguments. Participants are required to produce the annotations for events in the testing set. This involves producing event trigger annotations and event annotations by associating the annotated proteins and event triggers to an event.

5.2 Subgraph Matching-based Event Extraction

In this section, we formally describe a graph matching-based method to extract biological events from the biological literature in tackling the primary task of the BioNLP shared task. A *graph* is a pair of sets $G = (V, E)$ where V is a set of nodes and E is

a set of edges that connect pairs of nodes. Each edge is an ordered pair of nodes, i.e., $E \subseteq V^2$. A graph may be undirected, meaning that there is no distinction between the two nodes associated with each edge, or its edges may be directed from one node to another. A graph $S = (V', E')$ is a *subgraph* of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. We use notation $S \subseteq G$.

In this section, we first introduce the dependency representation and define the dependency graph which serves as the basis for our event extraction approach. We then propose an event rule induction method to automatically learn rules for detection of biological events from the dependency graphs of training sentences based on the gold event annotation. In the end, we describe our sentence matching approach that uses subgraph matching to match the event rules with each testing sentence in order to extract events in the sentence.

5.2.1 Dependency Representation

The dependency representation is designed to provide a simple description of the grammatical relationships in a sentence that can be effectively used to extract textual relations [91]. The dependency representation is often used in information extraction, including applications in the biological domain [4, 47, 93–95]. It was evaluated as the representation scheme for different natural language parsers to unify syntactic annotations of biological corpora [93, 94]. It was also used for extracting interactions between genes and proteins from the scientific literature [4, 47, 95].

The dependency representation for a sentence is formed by words or tokens in the sentence and the relationships between tokens. A single dependency relation is represented as $relation(governor, dependent)$ where *governor* and *dependent* are tokens in the sentence, and *relation* is the abbreviated grammatical dependency relation between the pair of tokens, which is determined based on existing relation hierarchies.

Since the dependency representation characterizes all sentence relationships uniformly as typed dependency relations between pairs of tokens, it can be presented as a directed graph, which is named *dependency graph* and defined in this work as follows:

Definition 5.2. (Dependency Graph) A dependency graph is a pair of sets

$G = (V, E)$ where V is a set of nodes that correspond to the tokens in a sentence, and E is a set of directed edges for which the edge labels correspond to grammatical dependency relations between the tokens, and the direction exists pointing from the *governor* token to the *dependent* token.

The graphs discussed in the subsequent sections all refer to the dependency graphs. Figure 5.2 illustrates the dependency representation and the dependency graph for the sentence: "Interferons inhibit activation of STAT6 by interleukin 4 in human monocytes by inducing SOCS-1 gene expression." (MEDLINE: 10485906). The token number in the sentence is appended to each token in order to differentiate identical tokens that co-occur in a sentence. All the protein names in the sentence have been replaced with a unified tag "BIO_Entity". The POS tag of each token is noted. "BIO_Entity" tokens are uniformly tagged as a proper noun.

5.2.2 Event Rule Induction

The premise of this work is that there is a set of frequently occurring event rules that match a majority of stated events about molecular biology. We consider that a biological event rule encodes the detailed description and the typical contextual structure of a group of biological events. In this work, a *biological event rule* is defined as follows:

Definition 5.3. (Biological Event Rule) A biological event rule is a pair $r = (f, G_r)$. $G_r = (V_r, E_r)$ is a dependency graph, which characterizes the contextual structure of events. $f = (\mathit{Type}, \mathit{Trigger}, \mathit{Arguments})$ encodes a detailed event frame, where *Type* is the event type, $\mathit{Trigger} = \{(t_1, v_1), (t_2, v_2), \cdots\}$ records the event trigger and is a non-empty finite sequence of tokens associated with nodes in G_r, i.e., $\mathit{Trigger} \in (W \times V_r)^+$, and $\mathit{Arguments} = \{(t_1, l_1, v_1), (t_2, l_2, v_2), \cdots\}$ records the event arguments and is a non-empty finite sequence of tokens associated with semantic role labels and nodes in G_r, i.e., $\mathit{Arguments} \in (W \times L \times V_r)^+$.

The event rule induction method that we propose to automatically learn biological event rules from training sentences is based on the use of gold event annotation.

Tokenized & Protein-replaced Sentence:

```
Interferons inhibit activation of BIO_Entity by
BIO_Entity in human monocytes by inducing BIO_Entity
gene expression .
```

Dependency representation:

```
nsubj(inhibit-2/VBP, Interferons-1/NNS)
dobj(inhibit-2/VBP, activation-3/NN)
prep_of(activation-3/NN, BIO_Entity-5/NNP)
prep_by(BIO_Entity-5/NNP, BIO_Entity-7/NNP)
amod(monocytes-10/NNS, human-9/JJ)
prep_in(BIO_Entity-7/NNP, monocytes-10/NNS)
prepc_by(inhibit-2/VBP, inducing-12/VBG)
nn(expression-15/NN, BIO_Entity-13/NNP)
nn(expression-15/NN, gene-14/NN)
dobj(inducing-12/VBG, expression-15/NN)
```

Dependency graph:

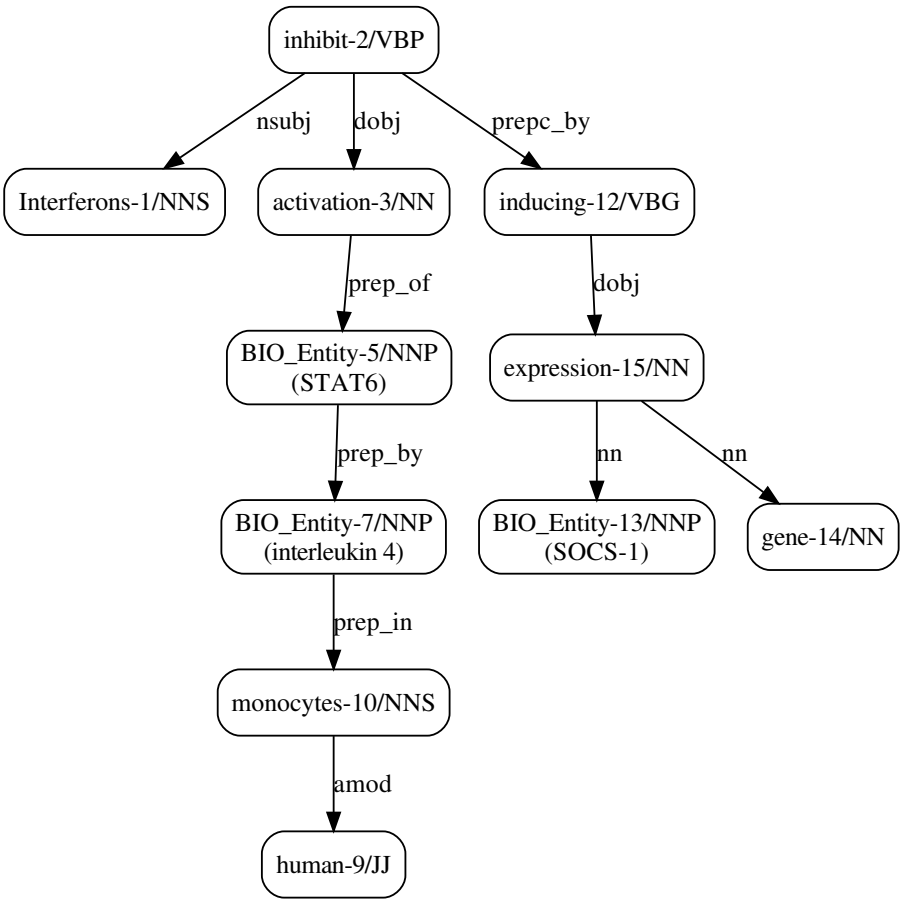


Figure 5.2: Dependency Representation and Dependency Graph Example

For the dependency graph of each training sentence, the directions of edges are first removed so the directed graph is transformed into an undirected graph, in which every edge can be traveled in either direction, and therefore a path must exist between any two nodes since the graph is always connected. For each gold event, the shortest dependency path in the undirected graph connecting the event trigger nodes to each event argument node is extracted. The union of all the shortest dependency paths is then computed for each event. In the end, the original directed dependency representation on the path union is retrieved and used as the graph representation of the event.

For multi-token event triggers, the shortest dependency path connecting the node of every trigger token to the node of each event argument is extracted, and the union of the paths is then computed for each trigger. For regulation events, when a sub-event is used as an argument, only the type and the trigger of the sub-event are preserved as the argument of the main events. The shortest dependency path is extracted so as to connect the trigger nodes of the main event to the trigger nodes of the sub-event. In case that there exists more than one shortest path, all of the paths are considered. As a result, each gold event is transformed into the form of a biological event rule. Algorithm 4 shows the details of inducing event rules from the gold events of each training sentence. The obtained biological event rules are categorized in terms of the nine event types of the shared task.

Figure 5.3 presents the undirected graph of the sentence exemplified in the Figure 5.2 and an event rule built from one of the gold events (Positive_regulation E1) of the sentence following the procedure of Algorithm 4. The detailed description and the dependency graph of the event rule are separated by an arrow. The tokens regarding proteins and event triggers in the sentence have been attached to the gold event annotation. The nodes involved in the dependency graph of the generated biological event rule are highlighted in the sentence graph.

5.2.3 Sentence Matching

We propose a sentence matching approach to attempt to match event rules to each testing sentence. Since the event rules and the sentences all possess a dependency graph, the matching process is a subgraph matching problem, which corresponds to

Original Sentence:

```
Interferons inhibit activation of STAT6 by interleukin 4
in human monocytes by inducing SOCS-1 gene expression.
```

Gold event E1:

```
ID   entity_type offset-begin offset-end text_span    token
T1   Protein      34           39         STAT6       (BIO_Entity-5/NNP)
T2   Protein      43           56      interleukin 4  (BIO_Entity-7/NNP)
T3   Protein      88           94         SOCS-1      (BIO_Entity-13/NNP)
T4   Positive_regulation   20   30      activation     (activation-3/NN)

ID   event_type       event_trigger     argument_1       argument_2
E1   Positive_regulation:    T4         Theme:T1         Cause:T2
```

Event rule of E1:

```
E1:    Positive_regulation:(activation-3/NN)
Theme:(BIO_Entity-5/NNP) Cause:(BIO_Entity-7/NNP)  <==
prep_by(BIO_Entity-5/NNP, BIO_Entity-7/NNP);
prep_of(activation-3/NN, BIO_Entity-5/NNP)
```

Undirected dependency graph:

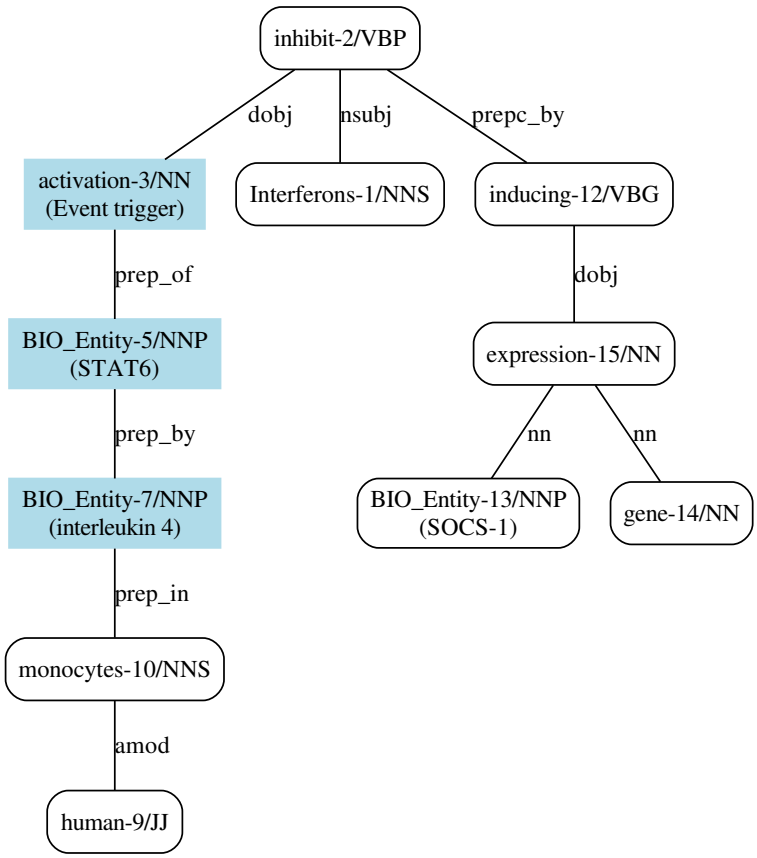


Figure 5.3: Event Rule Induction Example

Algorithm 4 Biological Event Rule Induction Algorithm

Input: Dependency graph of a training sentence s, $G_s = (V_s, E_s)$ where V_s is the set of nodes and E_s is the set of edges of the graph; a finite set of gold biological events that appear in s, $E_v = \{e_1, e_2, \cdots, e_i, \cdots\}$, where $e_i = (\textit{Type},\ \textit{Trigger},\ \textit{Arguments},\ s)$.

Output: A finite set of biological event rules $R = \{r_1, r_2, \cdots, r_i, \cdots\}$, where $r_i = (f_i,\ G_{r_i})$. $G_{r_i} = (V_{r_i},\ E_{r_i})$ is the dependency graph of r_i.

$R \leftarrow \emptyset$
for all $e_i \in E_v$ **do**
  $uG_s \leftarrow$ Undirected(G_s)
  //Undirected() transforms the directed graph G_s into an undirected graph uG_s
  $Path \leftarrow \emptyset$ // the initial Path set is empty
  for all *argument* $\in e_i.Arguments$ **do**
    $Path \leftarrow Path \cup \{$ Shortest_path($uG_s,\ e_i.Trigger,\ argument$) $\}$
    //Shortest_path() finds the shortest path(s) between trigger and argument
    //in the graph uG_s
  $G_{r_i} \leftarrow$ directed($G_s,\ Path$) //directed() retrieves the original directed
  //dependencies on the *Path* and generates the dependency graph G_{r_i}
  $f_i \leftarrow (e_i,\ G_{r_i})$
  $R \leftarrow R \cup \{\ r_i = (f_i,\ G_{r_i})\ \}$
return R

the search for a subgraph isomorphic to an event rule graph within the graph of a testing sentence. The subgraph matching problem is also called subgraph isomorphism [96]. Given an event rule graph $G_r = (V_r, E_r)$ and a sentence graph $G_s = (V_s, E_s)$, the subgraph isomorphism problem investigates if a graph G_r is isomorphic to a subgraph of G_s. If the answer is yes, a subgraph S_s of G_s to which G_r is isomorphic, as well as a bijective mapping of G_r to S_s which defines the isomorphism, will be provided. In this work, the *subgraph isomorphism* problem is defined as follows:

Definition 5.4. (Subgraph Isomorphism) An event rule graph $G_r = (V_r, E_r)$ is isomorphic to a subgraph of a sentence graph $G_s = (V_s, E_s)$, denoted by $G_r \cong S_s \subseteq G_s$, if there is an injective mapping $f : V_r \rightarrow V_s$ such that, for every directed pair of

nodes $v_i, v_j \in V_r$, if $(v_i, v_j) \in E_r$ then $(f(v_i), f(v_j)) \in E_s$, and the edge label of (v_i, v_j) is the same as the edge label of $(f(v_i), f(v_j))$.

Graph matching is a computationally expensive procedure. It has been proved that the subgraph isomorphism problem is NP-complete [97]. Therefore, it would be hard, and currently practically impossible, to find a polynomial, efficient algorithm for this problem. A number of algorithms has been designed to tackle the problem of subgraph isomorphism in different applications [98–104]. One of the well known methods was proposed by Ullmann based on the depth-first backtracking search with a forward checking procedure which greatly reduces the number of backtracking steps [98]. Considering that the graphs of rules and sentences involved in our matching process are small, a simpler subgraph matching algorithm using a backtracking approach is appropriate. The algorithm is named "Injective Graph Embedding Algorithm" and designed based on the Huet's graph unification algorithm [105]. The main algorithm and the recursive part of the algorithm are formalized in Algorithm 5 and Algorithm 6 respectively.

In the main algorithm, line 1 initializes the set *MR* to the empty set. For each event rule, line 3 finds the start node of the rule graph and line 5 determines the start nodes of the sentence graph. Each rule is only allowed to have one start node while each sentence can possess a set of start nodes.

Two scenarios are considered in the algorithm in finding the start nodes. First, if the rule contains at least one "BIO_Entity" token, the "BIO_Entity" token that has the lowest token number becomes the start node of the rule. For instance, "BIO_Entity-3/NNP" becomes the start node if the rule contains the two "BIO_Entity" tokens: "BIO_Entity-3/NNP" and "BIO_Entity-6/NNP". This does not reduce the set of found solutions. In the meantime, every "BIO_Entity" token in the sentence becomes an alternate start node for the sentence. Second, if the rule does not have any "BIO_Entity" token, the token with the lowest token number becomes the start node of the rule, while every token in the sentence becomes one of the start nodes. The second scenario applies especially to the event rules that are obtained from regulation events that take only sub-events as their primary arguments. In this case, it is quite likely that there is no "BIO_Entity" token involved on the shortest dependency path

Algorithm 5 Injective Graph Embedding Algorithm (Main algorithm)

Input: Dependency graph of a testing sentence s, $G_s = (V_s, E_s)$ where V is the set of nodes and E is the set of edges of the graph; a finite set of biological event rules $R = \{r_1, r_2, \cdots, r_i, \cdots\}$, where $r_i = (e_i,\ G_{r_i})$. $G_{r_i} = (V_{r_i},\ E_{r_i})$ is the dependency graph of r_i.

Output: MR : a set of biological event rules from R matched with s together with the injective mapping

Main algorithm:

```
1:  MR ← ∅
2:  for all r_i ∈ R do
3:      st_{r_i} ← StartNode(G_{r_i})
4:      //StartNode finds the start node st_{r_i} of the rule graph G_{r_i}
5:      ST_s ← {st_{s_1}, st_{s_2}, ···, st_{s_j}, ···}
6:      //ST_s : the set of start nodes of the sentence graph G_s
7:      for all st_{s_j} ∈ ST_s do
8:          create an empty stack σ and push (st_{r_i}, st_{s_j}) onto the stack σ
9:          IM ← ∅  //IM : records of injective matches between nodes in G_{r_i} and G_s
10:         call MatchNode(σ, rIM, G_{r_i}, G_s)        //rIM : reference of IM
11:         if MatchNode() returned TRUE then
12:             MR ← MR ∪ {r_i with IM }
13: return MR
```

connecting the trigger nodes of the main event to the trigger nodes of the sub-event.

The *for* loop of lines 7–12 attempts to match the rule graph to the sentence graph, starting from matching the start node of the rule graph with each start node of the sentence graph via the recursive subroutine MatchNode. By allowing the rule to match the sentence at different positions, the algorithm is capable to extract all the underlying biological events in the sentence.

In the recursive subroutine MatchNode, line 1 initializes the current-level match records between nodes in the rule graph and the sentence graph to the parent-level match records. In the *while* loop of lines 2–27, lines 4–5 check if an injective match exists between the current rule node v_r and the current sentence node v_s. In determining an injective match between two nodes, the algorithm can compare the

Algorithm 6 Injective Graph Embedding Algorithm (Recursive subroutine)

Recursive subroutine: MatchNode($\sigma, rIM_{parent}, G_{r_i}, G_s$)

```
1:  IM_current ← IM_parent   //assign IM_parent from the parent level to the current IM_current
2:  while stack σ is not empty do
3:      pop node pair (v_r, v_s) from stack σ
4:      if an injective match between v_r and v_s already exists in IM_current then
5:          do nothing
6:      else if an injective match is possible between v_r and v_s then
7:          IM_current ← IM_current ∪ { the match between v_r and v_s }
8:      else
9:          return FALSE
10:     for all edges e_r adjacent to node v_r in G_{r_i} do
11:         let (v_r, n_r) be the edge e_r
12:         for all edges e_s adjacent to node v_s in G_s do
13:             let (v_s, n_s) be the edge e_s
14:             if e_r and e_s share a same direction and possess identical edge labels then
15:                 S ← S ∪ n_s                    //S : the set of candidate nodes for matching n_r
16:         for all n_s ∈ S do
17:             if an injective match between n_r and n_s already exists in IM_current then
18:                 go to Line 10 and proceed with next edge e_r
19:             else if an injective match is possible between n_r and n_s then
20:                 σ_n ← σ                        //copy σ to a new stack σ_n
21:                 push (v_r, v_s, n_r, n_s) onto the stack σ_n
22:                 call MatchNode(σ_n, rIM_current, G_{r_i}, G_s)
23:                 //rIM_current : reference of IM_current
24:                 if MatchNode() returned TRUE then
25:                     IM_parent ← IM_current     //update IM_parent using IM_current
26:                     return TRUE
27:         return FALSE
28: IM_parent ← IM_current
29: return TRUE
```

corresponding tokens and their attached POS tags of the nodes. The attached token number indicates only the relative position of a token in its original sentence, and so is not used in matching. Therefore, nodes involving protein names can always match with each other because they are represented by the unified tag "BIO_Entity" and possess the same POS tag NNP.

If an injective match is created between v_r and v_s, for each edge e_r adjacent to v_r, line 14 checks if there exists an edge e_s adjacent to v_s that has the same direction and the same edge label as e_r. The two edges can match only if they share the same direction and the edge labels are identical. In terms of each matched edge e_s, line 15 retrieves a set of nodes that appears on the other end of e_s as the candidate nodes to match with the node n_r on the other side of e_r. S is the set of candidate nodes, which will be used in backtracking. For each candidate node n_s, line 19 checks if an injective match is possible between n_s and n_r. If the match does not exist, the algorithm attempts to match n_r with the next candidate node n_s through line 16. If the match exists between n_s and n_r, lines 20–22 push the nodes v_r, v_s, n_r and n_s for future matching, invoke the subroutine MatchNode recursively, and repeat the matching process for these nodes in the next level. In line 21, the parent node pair (v_r, v_s) of (n_r, n_s) is also passed with the node pair (n_r, n_s) to the next level to make sure that every child node of v_r has an injective match with a child node of v_s before signaling the parent level the success of matching in the current level. If MatchNode returns $TRUE$, lines 25–26 update the parent-level match records using the current-level match records, and return to the parent-level program with $TRUE$. Otherwise, the algorithm backtracks and attempts to match n_r with the next candidate node n_s via line 16.

An example of the backtracking process when matching a rule graph with a sentence graph is illustrated in Figure 5.4. The matches are highlighted by dotted lines. As shown in (a), the algorithm finds the injective match between the A nodes in the rule graph and the sentence graph, and then proceeds to detect the injective matches 1 between B nodes and 2 between C nodes. Before signaling the parent level the success of the matches of nodes in the right branch of the node A in the rule graph, the algorithm checks if the nodes in the left branch can also find their injective matches in the sentence graph based on the available parent node pair information (v_r, v_s).

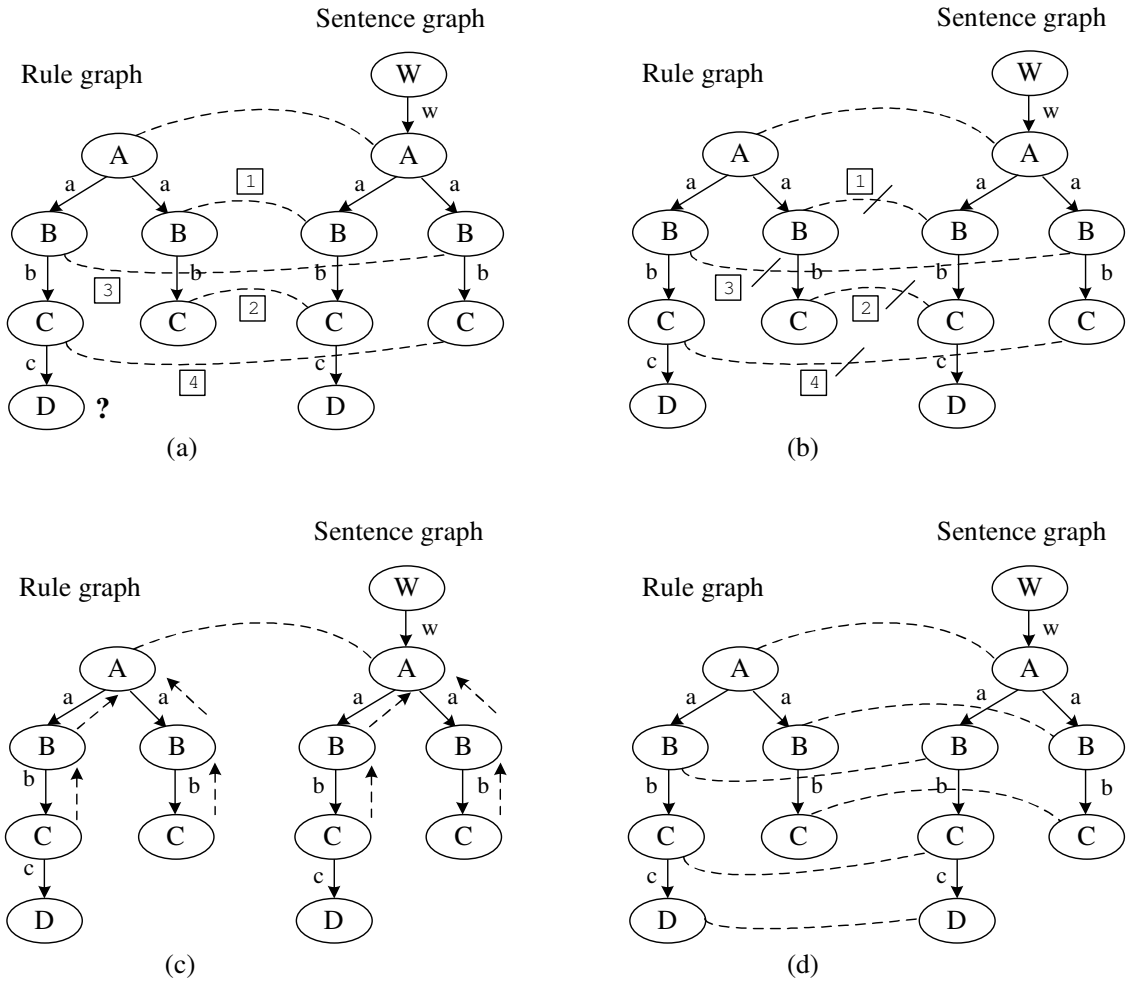


Figure 5.4: Example of Backtracking Process (a) initial injective matches (b) wrong matches detected (c) backtracking to the A node (d) correct matches found

The algorithm further detects the injective matches 3 and 4. However, the node D in the left branch of the node A in the rule graph does not have its injective match in the sentence graph. (b) The algorithm detects the current matches 1 between B nodes and 2 between C nodes cannot guarantee that every node in the left branch of the node A in the rule graph find their injective matches in the sentence graph. (c) The algorithm gives up the matches between B nodes and C nodes, backtracks to the A nodes, and attempts to match the B node in the right branch of the node A in the rule graph with the other candidate node in the sentence graph. (d) The algorithm finds the correct matches leading to a subgraph matching between the rule graph and the sentence graph.

The complexity of Algorithm 5 (Injective Graph Embedding Algorithm) is exponential, as we could expect since the problem of subgraph isomorphism is known to be NP-hard. However, we observed that the algorithm is relatively efficient in practice and we have successfully run it on the BioNLP'09 shared task. We explain here why this efficient performance in practice can be expected. Let us assume that the sentence dependency graph G_s and the rule graph G_{r_i} have a total of n vertices and m edges, and the vertex degree (number of adjacent edges) is always less than or equal to k. The main algorithm has two nested loops so it calls the recursive part MatchNode $O(|R| \cdot n)$ times. When calling the recursive part, MatchNode, the main source of inefficiency is the occurrence of several edges with the same label, adjacent to one node. This is more an exception than the rule. If we had two graphs with no adjacent same-label edges, a Huet-style algorithm would test for matching in practically linear time $O(n + m)$, which would give $O(|R| \cdot n(n + m))$ total running time. However, in our case, if the two graphs do not have same-labeled, adjacent edges, MatchNode would be called for each pair of matchable nodes, which makes $O(n)$ invocations. The nested loops iterate $O(nk^2)$ times. Since line 20 requires $O(n)$ time to copy the stack, the total time would be $O(|R| \cdot n^3k^2)$ time. However, if same-label adjacent edges are present, the algorithm may backtrack to try to match each edge with k possible edges in the other graph, which gives $O(k^n)$ possible invocations of MatchNode, with the total worst-case algorithm complexity $O(|R| \cdot n^2k^n)$.

According to the GENIA corpus, on average there are about 24 words in each sentence [21], which correspond to the nodes in the dependency graph. Therefore, the input graphs of sentences and rules are not large graphs. In addition, by matching from pairs of start nodes, the search can be efficiently narrowed. In practice, it only takes the algorithm a couple of seconds to return the results. Hence, our algorithm is efficiently solving the subgraph matching problem in this work.

The algorithm proceeds until a subgraph isomorphic to the rule graph is found in the sentence graph. For each sentence, the algorithm returns all the matched rules together with the corresponding injective mappings from rule nodes to sentence tokens. Biological events in the testing sentence are then extracted using the event descriptions of tokens in each matched rule consisting of the type, the trigger and the arguments to the corresponding tokens of the sentence. Figure 5.5 presents a simple

example of the event extraction process by matching an event rule to a sentence to extract a Positive_regulation event in the sentence. The matching criteria in the example require that edges be matched if they share a same direction and possess identical edge labels while nodes be matched as long as the POS tags of the tokens are the same.

The subgraph isomorphism problem defined here can also be adapted to determine the relationship between any pair of event rules. The detailed application is presented in Section 5.4.2.

5.3 Implementation

Figure 5.6 illustrates the overall architecture of our event extraction process, which consists of four major components: preprocessing, rule induction, sentence matching and postprocessing.

5.3.1 Preprocessing

In our work, several standard preprocessing steps are completed on both training and testing data before performing text mining strategies. These include sentence segmentation and tokenization, Part-of-Speech tagging, and sentence parsing.

Sentence segmentation and tokenization

We assume that a sentence is the suitable granularity degree in event extraction. Therefore, the target text is first segmented into sentences. Then, each sentence is tokenized with whitespace separating tokens. These processes are done using tools integrated into U-compare, an integrated text mining system based on the UIMA Framework [106]. Penn Treebank escapes, e.g., "-LRB-" for "(", are used in the tokenization. Since the gold annotation of proteins or events is provided based on the entire target text, the annotation is then divided and attached to each tokenized sentence. For training data, only sentences that contain at least one protein and one event are considered for further processing. For testing data, sentences that contain no protein names are removed. Next, we require that every protein be separated from surrounding texts and become one individual token. For instance, the word

Input:

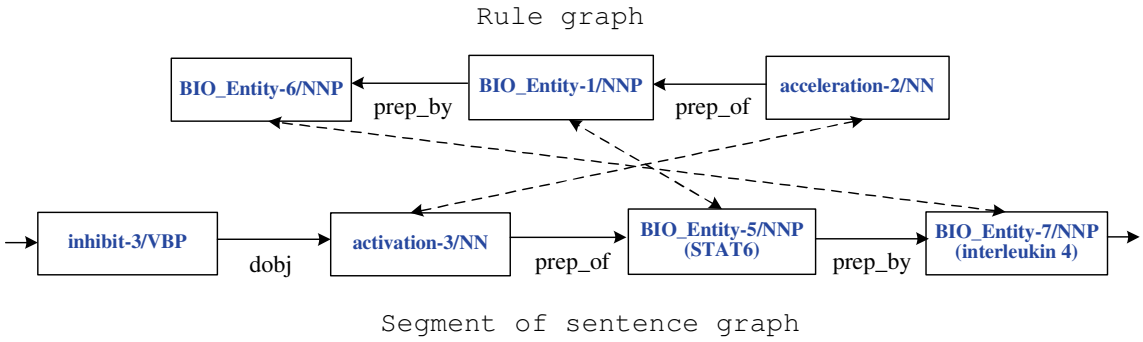
Event rule:

```
Positive_regulation:(acceleration-2/NN)
Theme:(BIO_Entity-1/NNP) Cause:(BIO_Entity-6/NNP)  <==
prep_by(BIO_Entity-1/NNP, BIO_Entity-6/NNP);
prep_of(acceleration-2/NN, BIO_Entity-1/NNP)
```

Sentence:

```
Interferons inhibit activation of STAT6 by interleukin 4
in human monocytes by inducing SOCS-1 gene expression.
```

Subgraph matching:



Injective mapping:

Rule			Sentence
BIO_Entity-1/NNP	(Theme)	←-→	STAT6
BIO_Entity-6/NNP	(Cause)	←-→	interleukin 4
acceleration-2/NN	(trigger)	←-→	activation-3/NN

Extracted event:

```
Positive_regulation:(activation)  Theme:(STAT6)
Cause:(interleukin 4)
```

Figure 5.5: Event Extraction Example

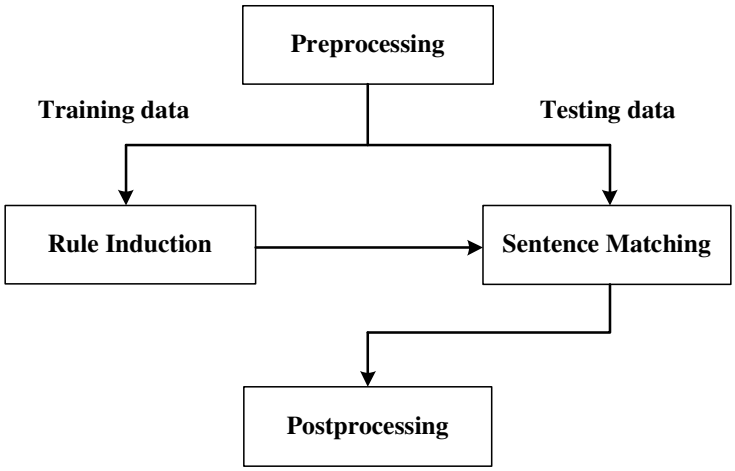


Figure 5.6: General Architecture of Event Extraction

"HIV-TF1-binding" forms two tokens "HIV-TF1" and "-binding" as "HIV-TF1" is a protein name.

Biological sentences are generally complex because occurrences of multi-word named entities are common in biological texts. First, such entities are not likely to appear in the dictionary of a general English parser, and will force the parser to use morpho-guessing and unknown word guessing. This is time consuming and prone to error. Second, such entities consist of many words that have various morphological tags. One named entity can be divided into different phrases. This causes the sentence structure to collapse, and thus makes biological texts difficult to parse.

It has been demonstrated in recent research publications [5, 107] that substituting named entity with one predefined word effectively simplifies the structures of biological sentences, reduces ambiguities arising from complex sentences, and thus improves both the accuracy and efficiency of the parsing process in the later stages. Therefore, in our work all the protein names are replaced with a unified tag "BIO_Entity". For instance, the five-token protein name "cAMP response element binding protein" then becomes one individual token "BIO_Entity".

Part-of-Speech tagging

POS tagging is performed on the tokenized sentences to associate each word in a sentence with its most likely Part-of-Speech tag. Because subsequent processing steps typically depend on the tagger's output, high accuracy at this level is crucial for

success in later phases. Therefore, a domain specific tagger, GENIA tagger [108], is used to perform this task. It is assumed that the parser can produce better parsing results if it receives the results from an in-domain tagger instead of its embedded tagger for general English. The reported POS tagging accuracy of GENIA tagger on the GENIA corpus is 98.55% [108]. We have tuned GENIA tagger to tag the token "BIO_Entity"as a proper noun NNP.

Sentence parsing

The POS-tagged sentences are then submitted to the Stanford unlexicalized natural language parser [109] to analyze the syntactic and semantic structure of the sentences. Some work has been done on evaluating the performance of state-of-the-art parsers on general English [110, 111]. The Stanford parser achieves a 84.2% F-score of labeled attachment on section 22 of the Penn TreeBank, and is reported to be one of the best parsers in terms of speed and accuracy [111]. The Stanford parser has also been successfully applied to the biological domain. It achieves top performance on the task of extracting protein-protein interactions from the biological literature [64], and helps to identify correct patterns for constructing a comprehensive dictionary of medical treatment terms from randomized clinical trial (RCT) reports [112].

The Stanford parser returns the dependency representation for each sentence. The current representation contains a hierarchy of 55 grammatical relations. The most generic relation, dependent (dep), will be used when a more precise relation in the hierarchy does not exist or cannot be retrieved by the parser.

5.3.2 Rule Induction

For each gold event, the shortest dependency path in the undirected graph connecting the event trigger to each event argument is extracted using Dijkstra's algorithm [113]. Dijkstra's algorithm is an efficient algorithm for finding the shortest path between any two nodes in a weighted graph. In this work, we assign each edge an equal weight.

5.3.3 Sentence Matching and Postprocessing

Sentence matching is performed following the procedure of Algorithm 5 and Algorithm 6. Some postprocessing rules are then applied to the raw sentence matching

results to produce the event annotations in the right format for evaluation by the shared task. All the applied rules in our work have been defined in the specifications of the BioNLP shared task [1]:

- Trigger cannot be a protein name or an event.
- Only a protein or an event can be a theme or a cause for regulation events.
- Only a protein can be a theme for non-regulation events.
- Duplicate events should be removed.
- All regulation events involving sub-events that are not extracted should be removed.

5.4 Results and Evaluation

This section starts with the description of preprocessing results on the datasets of the BioNLP'09 shared task. Event rule induction results are then presented. Next, the results of three different experimental methods in processing the obtained event rules are described in detail. In the end, the evaluation results on the extracted biological events based on the evaluation measures of the shared task are reported and discussed. Compared to the reported results of the 24 participating teams of the shared task [1], the official evaluation shows that our results would rank 6th in extracting biological events in the testing data.

5.4.1 Preprocessing Results

For the training dataset which consists of 800 biological research abstracts, there are 8,597 biological events according to the gold event annotation. Among them, 946 events involve proteins or sub-events that span more than one sentence. Our proposed graph matching-based method focuses on extracting biological events from sentences. Therefore, only sentence-based events will be considered in this work. The remaining 7,651 events all occur in individual sentences and are distributed in the 3,036 candidate training sentences, each of which contains at least one protein and one event. For the development dataset, 988 candidate sentences are extracted,

each of which contains at least one protein. For the testing dataset, 1,670 candidate sentences are extracted using the same criterion. Table 5.2 presents some statistics of the preprocessed datasets.

Attributes	Training set	Development set	Testing set
Abstracts	800	150	260
Total sentences	7,482	1,450	2,448
Candidate sentences	3,036	988	1,670
Total events	8,597	1,789	hidden
Sentence-based events	7,651	1,639	hidden

Table 5.2: Statistics of experimental datasets

5.4.2 Rule Induction Results

A dependency graph is built for each preprocessed sentence based on the default typed dependency representation of the Stanford parser, namely, collapsed dependencies with propagation of conjunct dependencies [91]. We were able to build biological event rules for 7,544 gold events. Gold events in which the event trigger and an event argument are not connected by a path in the undirected dependency graph of the sentence could not be transformed into a biological event rule. After removing duplicate rules, we obtained 6,435 event rules, which are distributed over nine predefined event types of the shared task shown in Table 5.3.

Instead of directly matching the biological event rules to each of the preprocessed sentences in the development and testing sets, we processed the rules of each event type via three different experimental methods for the purpose of refining the obtained event rules. The three resulting rule sets are: partial order rule set, all rule set and non-overlapping rule set. The detailed results are presented as follows, which demonstrate that the non-overlapping rule set is more appropriate to become our final choice of rule set.

Partial order rule set

In order theory, a partial order set (poset) is defined as a set together with a binary relation that is reflexive, antisymmetric, and transitive [114]. For certain pairs of

Event type	Number of event rules
Gene_expression	1,163
Transcription	461
Protein_catabolism	91
Phosphorylation	139
Localization	223
Binding	774
Regulation	734
Positive_regulation	2,125
Negative_regulation	725
Grand total	6,435

Table 5.3: Event rule distribution

elements in the set, one of the elements precedes the other. For some pairs, neither element precedes the other in the set since not every pair of elements of a poset needs to be related. Based on the graph representation of each event rule, a partial order rule set is built for each event type, in which for certain pairs of rules, the more general rule precedes the more specific rule.

First, the proposed injective graph embedding algorithm is performed for each pair of rules within the event type by matching the dependency graphs of two rules, rule A and rule B, with each other. Three independent resulting scenarios are considered: (1) if the graph of rule A is isomorphic to a subgraph of the graph of rule B, and vice versa, rule A is then considered isomorphic to rule B. We keep only one rule for the event type and randomly remove the other from the poset. (2) if the graph of rule A is isomorphic to a subgraph of the graph of rule B, but the graph of rule B is not isomorphic to any subgraph of the graph of rule A, rule A is then considered more general than rule B. We relate rule A with rule B in the poset with rule A preceding rule B. (3) if the graph of rule A is not isomorphic to any subgraph of the graph of rule B, and vice versa, rule A is then considered independent of rule B. We insert both rule A and rule B into the poset but neither rule precedes the other. As a result, the general-specific relation between the dependency graphs of certain event rules in the built poset is reflexive, antisymmetric, and transitive.

When matching between rule graphs, different combinations of matching features are applied, resulting in different partial order rule sets. The features include edge

features (Edge) which are edge label and edge direction, and node features which are POS tags (POS), trigger tokens (Trigger), and all tokens (All), ranging from the least specific matching criterion, Edge, to the much stricter criterion, All. Table 5.4 shows the event rule distribution of one of the resulting partial order rule sets based on the combination of 3 matching features, edges, POS tags and trigger tokens, denoted by "Edge+POS+Trigger", which means that the edge directions and labels, the POS tags of tokens and the event trigger tokens have to be exactly the same for the edges and the nodes of two rules to match with each other. Compared to the statistics in Table 5.3, the number of event rules is reduced by some degree for every event type due to the removal of the isomorphic rules, as described in the scenario (1).

Event type	POset	Original set	Reduction Percentage
Gene_expression	522	1,163	55%
Transcription	241	461	48%
Protein_catabolism	46	91	49%
Phosphorylation	68	139	51%
Localization	136	223	39%
Binding	559	774	28%
Regulation	516	734	30%
Positive_regulation	1,410	2,125	34%
Negative_regulation	512	725	29%
Grand total	4,010	6,435	38%

Table 5.4: Distribution of distinct event rules of poset (Edge+POS+Trigger)

Next, the proposed injective graph embedding algorithm is performed again between the rules in each of the resulting posets and the sentences in the training set in order to extract biological events from training sentences. For each sentence, rules are matched with it respectively following the inherent order of the poset. Instead of returning all the matched rules, only the most specific rules among them are preferred, i.e., the matched rules to which there is no matched rule in the poset more specific. We have experimented with all the posets in terms of different combinations of matching features. Table 5.5 gives the results of the event extraction on training sentences using the partial order rule set based on "Edge+POS+Trigger".

The performance is evaluated using "Approximate Span Matching/Approximate

Recursive Matching", the primary evaluation criterion of the shared task. The *approximate span matching* is defined by relaxing the requirement for text span matching for identified event trigger expressions. Specifically, a given span is equivalent to a gold span if it is entirely contained within an extension of the gold span by one word both to the left and to the right. The *approximate recursive matching* is defined by relaxing the requirement for recursive event matching, so that an event can match even if the events it refers to are only partially correct. Specifically, events can match even if referred events differ in non-Theme arguments.

Event type	Precision(%)	Recall(%)	F-score(%)
Gene_expression	73.06	90.41	80.81
Transcription	29.73	86.60	44.26
Protein_catabolism	92.45	95.15	93.78
Phosphorylation	92.99	93.59	93.29
Localization	22.85	89.84	36.43
Binding	82.35	84.00	83.17
Regulation	69.83	63.01	66.24
Positive_regulation	58.62	61.12	59.84
Negative_regulation	68.31	58.16	62.83
All total	57.93	73.30	64.71

Table 5.5: Event extraction of poset (Edge+POS+Trigger) on training sentences evaluated by Approximate Span Matching/Approximate Recursive Matching

Since the event rules are induced originally from the gold events that appear in the training sentences, ideally, mapping event rules back to the training sentences will retrieve all the gold biological events, i.e., 100% recall. According to the results in Table 5.5, the recalls of the first 6 event types including Binding still are less than 100%, while the recalls of regulation event types are only in the 60% range. By investigating the biological events that cannot be extracted from the training sentences, two problems are observed with regard to the obtained partial order rule sets. We will illustrate the two observed issues using two examples.

(1) Suppose we have the following two Binding event rules:

rule A:

Binding:(binding-11/VBG) Theme:(BIO_Entity-15/NNP) $\Longleftarrow$

prep_to(binding-11/VBG, BIO_Entity-15/NNP)

rule B:
Binding:(binding-11/VBG) Theme:(BIO_Entity-13/NNP) $\Longleftarrow$
nn(BIO_Entity-18/NNP, BIO_Entity-13/NNP);
prep_to(binding-11/VBG, BIO_Entity-18/NNP)

Rule A precedes rule B in the built Binding poset in terms of the "Edge+POS+Trigger" matching because rule A is more general than rule B according to the graph representation on the right hand side of the rules. Therefore, rule B is more specific and rule A will not be considered if both rules match with a sentence. However, the detailed event information about the two rules on the left hand side are actually quite different. For rule A, the event information describes Binding events involving the trigger "binding" and a protein that has a dependency relation with the trigger. For rule B, the event information discusses Binding events involving the trigger "binding" and a protein that has no direct dependency relation with the trigger. Therefore, returning only rule B will miss the potential events that rule A is able to extract from the sentence.

(2) Suppose we have the following two Regulation event rules:
rule A:
Regulation:(DNA-binding-1/JJ) Theme:(BIO_Entity-4/NNP) $\Longleftarrow$
amod(properties-2/NNS, DNA-binding-1/JJ);
prep_of(properties-2/NNS, BIO_Entity-4/NNP)
rule B:
Regulation:(DNA-binding-1/JJ) Theme:(Binding:properties-2/NNS) $\Longleftarrow$
amod(properties-2/NNS, DNA-binding-1/JJ);
prep_of(properties-2/NNS, BIO_Entity-4/NNP)

Only one of the rules will be kept in the Regulation poset in terms of the "Edge+POS+Trigger" matching because rule A is isomorphic to rule B according to the graph representation on the right hand side of the rules. For rule A, the event information describes Regulation events involving the trigger "DNA-binding" and a protein. However, for rule B, the event information describes Regulation events involving the trigger "DNA-binding" and a Binding sub-event whose trigger is "properties". Therefore, removing either rule will miss the potential events that the other rule can extract from a sentence. This explains why the recall is especially low for all

the regulation events.

All rule set

We consider using the set of all obtained event rules shown in Table 5.3. Table 5.6 shows the results of matching all the rules with the training sentences for event extraction based on the matching feature "Edge+POS+Trigger". Duplicate events are removed from the extracted events in the postprocessing step.

Event type	Precision(%)	Recall(%)	F-score(%)
Gene_expression	71.34	98.75	82.84
Transcription	25.04	96.60	39.77
Protein_catabolism	91.82	98.06	94.84
Phosphorylation	92.17	98.08	95.03
Localization	24.16	99.19	38.86
Binding	72.92	92.25	81.45
Regulation	56.31	88.67	68.88
Positive_regulation	51.31	86.12	64.31
Negative_regulation	57.37	82.54	67.69
All total	52.09	90.81	66.20

Table 5.6: Event extraction of all rule set on training sentences evaluated by Approximate Span Matching/Approximate Recursive Matching

The recalls of the first 6 event types are approaching 100%, while the recalls of regulation event types are a little lower but still at an average of about 86%, which we attribute to the accumulated errors in the first 6 event types that also serve as sub-events of the regulation events because incorrect sub-events will lead to wrong regulation events. However, the precision of Transcription and Localization is much lower than of other event types, staying only at the 20% level.

By investigating the false positives of the extracted biological events, an interesting problem is found. We observed that some event rules of Transcription and Localization are overlapped with rules of other event types. For instance, for the following two rules, the Transcription rule is isomorphic to the Gene_expression rule in terms of the graph representation and they also share a same event trigger token. In fact, tokens like "gene expression" and "induction" are used as event triggers

of both Transcription and Gene_expression in training data. Therefore, the detection of some Gene_expression events is always accompanied by certain Transcription events. This will have detrimental effect on the precision of both Transcription and Gene_expression event types.

Transcription is the process of creating an equivalent RNA copy of a sequence of DNA. According to the central dogma of molecular biology, transcription is the first step leading to gene expression [7]. Therefore, there exist some correlations or associations between the two event types. In tackling this problem, we processed the rules and built a non-overlapping rule set.

rule A:

Transcription:(express-10/VB) Theme:(BIO_Entity-11/NNP) $\Longleftarrow$
dobj(express-10/VB, mRNA-12/NN); nn(mRNA-12/NN, BIO_Entity-11/NNP)

rule B:

Gene_expression:(express-5/VB) Theme:(BIO_Entity-6/NNP) $\Longleftarrow$
dobj(express-5/VB, mRNA-7/NN); nn(mRNA-7/NN, BIO_Entity-6/NNP)

Non-overlapping rule set

When the dependency graphs of two rules across different event types are isomorphic to each other and two rules share a same event trigger token, we keep the rule of the event type in which the trigger token of the rule occurs more frequent as a trigger in the training data, and remove the rule of the other event type from the set. For the overlapping Transcription and Gene_expression rules in the above example, since the event trigger token "express" appear more often as trigger in Gene_expression than Transcription, the Transcription rule will be eliminated from the rule set. Table 5.7 displays the results of the event extraction on training sentences using the non-overlapping rule set based on "Edge+POS+Trigger".

Compared to the Table 5.6, the non-overlapping rule set helps to improve the overall precision by nearly 20% at a cost of about 4% drop on the overall recall. The precision of Localization is increased from 24% to 93%. We consider that this rule set is more appropriate for event extraction and therefore it becomes our final choice of rule set.

Event type	Precision(%)	Recall(%)	F-score(%)
Gene_expression	80.40	96.87	87.87
Transcription	56.08	76.60	64.75
Protein_catabolism	91.82	98.06	94.84
Phosphorylation	92.12	97.44	94.71
Localization	92.98	91.46	92.21
Binding	72.92	92.25	81.45
Regulation	67.02	84.94	74.92
Positive_regulation	65.70	81.78	72.86
Negative_regulation	70.42	78.00	74.02
All total	70.89	86.42	77.89

Table 5.7: Event extraction of non-overlapping set on training sentences evaluated by Approximate Span Matching/Approximate Recursive Matching

5.4.3 Event Extraction Results on Development Set

The non-overlapping rule sets in terms of different combinations of matching features are then applied respectively on the 988 candidate sentences in the development set using the proposed graph matching algorithm. The overall performance of the event extraction based on each feature is provided in Table 5.8.

The least specific matching criterion when matching between rules and sentences is "Edge", in which, without checking any information about nodes, as long as edge directions and labels are the same, both edges and nodes of a rule and a sentence can match with each other. It achieves the highest recall among all the runs and captures more than half of the gold events in the development sentences. However, the precision is quite low, leading to a very low F-score as too many false positives are generated due to disregard of node information.

As the strictest matching criteria, "Edge+POS+All" requires that the edges, the POS tags and all tokens be exactly the same for the edges and the nodes of a rule and a sentence to match with each other. It achieves the highest precision 69.72% and an F-score over 40%. This indicates that many biological events are actually described in very similar ways in the literature, involving the same grammatical structures and identical contextual contents. Comparing to the use of "POS+All", it is noticed that adding the edge features improves the overall precision of event extraction by a

large margin, nearly 13%. However, "POS+All" achieves a better recall due to the exclusion of edge labels, leading to a higher F-score. "Edge+POS+Trigger" requires that edge directions and labels of all edges be identical, POS tags of all tokens be identical, and tokens of only event triggers be identical. It achieves better performance than "Edge+POS+All" when relaxing the matching criteria from all tokens being the same to only event trigger tokens having to be identical. The best 2 of the first 6 runs in Table 5.8 are "Edge+POS+Trigger" and "POS+All", having an F-score of 40.63% and 41.42% respectively.

Next, we attempted to relax the matching criterion of POS tags for nouns and verbs. For nouns, the plural form of nouns is allowed to match with the singular form, and proper nouns are allowed to match with regular nouns. For verbs, past tense, present tense and base present form are allowed to match with each other. Further, the event trigger tokens are stemmed to their root forms using the Porter's stemming algorithm allowing the trigger tokens derived from a same root word to match [115]. "Edge+Relaxed_POS+Trigger_Stemming" and
"Relaxed_POS+All+Trigger_Stemming" in Table 5.8 demonstrate the improved performance to the above best two runs. These modifications improve the recall but produce many incorrect events, leading to only a small increase on the overall F-score.

Feature	Precision(%)	Recall(%)	F-score(%)
Edge	1.22	**52.26**	2.38
Edge+POS	2.23	45.33	4.25
Edge+POS+All	**69.72**	28.06	40.02
Edge+POS+Trigger	58.85	31.02	**40.63**
POS+All	57.00	32.53	**41.42**
POS+Trigger	40.65	36.95	38.71
Edge+Relaxed_POS+Trigger_Stemming	50.86	34.71	**41.26**
Relaxed_POS+All+Trigger_Stemming	51.51	35.22	**41.84**

Table 5.8: Event extraction of non-overlapping set on development set using different features evaluated by Approximate Span Matching/Approximate Recursive Matching

5.4.4 Event Extraction Results on Testing Set

According to the performance on the development set, we decided to conduct four runs on the testing sentences in terms of 4 features: "Edge", "Edge+POS+All", "Edge+Relaxed_POS+Trigger_Stemming" and "Relaxed_POS+All+Trigger_Stemming". For "Edge" and "Edge+POS+All", we aim to investigate the highest recall and precision on the testing sentences that can be achieved by our method. We submitted the results to the shared task website [92] to receive official evaluations. Table 5.9 gives the event extraction results on the 1,670 testing sentences using the non-overlapping rule sets in terms of the 4 features.

Feature	Precision(%)	Recall(%)	F-score(%)
Edge	0.84	**52.17**	1.65
Edge+POS+All	**58.64**	26.02	36.05
Edge+Relaxed_POS+Trigger_Stemming	41.77	33.66	**37.28**
Relaxed_POS+All+Trigger_Stemming	39.61	32.18	35.51

Table 5.9: Event extraction of non-overlapping set on testing sentences using different features evaluated by Approximate Span Matching/Approximate Recursive Matching

"Edge+Relaxed_POS+Trigger_Stemming" achieves the best overall F-score of 37.28% among all the runs. Similarly to the development set, the highest precision 58.64% on the testing sentences is achieved by the strictest matching criteria "Edge+POS+All". The highest recall 52.17% is obtained by the least specific matching criterion "Edge", indicating that a large amount of biological events are described in quite different grammatical structures in the literature. Although "Relaxed_POS+All+Trigger_Stemming" produced the best performance on the development set, it does not perform as well on the testing set. This clearly suggests that when requiring every token to be exactly the same for matching nodes of a rule and a sentence, the event rules have less stable generalization power to capture the underlying events. Table 5.10 shows the detailed results of the best run "Edge+Relaxed_POS+Trigger_Stemming". It can be seen that the performance of regulation events is much lower than other event types. This is not surprising because regulation events can have multiple arguments which might involve sub-events. For a regulation event to be detected successfully, the nested sub-events must be correctly

extracted first.

Event type	Precision(%)	Recall(%)	F-score(%)
Gene_expression	66.67	55.68	60.68
Transcription	31.37	35.04	33.10
Protein_catabolism	77.78	50.00	60.87
Phosphorylation	66.38	57.04	61.35
Localization	90.77	33.91	49.37
Binding	42.13	30.84	35.61
Regulation	23.32	15.46	18.60
Positive_regulation	27.09	25.33	26.18
Negative_regulation	30.56	20.32	24.41
All total	41.77	33.66	**37.28**

Table 5.10: Event extraction of "Edge+Relaxed_POS+Trigger_Stemming" on testing sentences evaluated by Approximate Span Matching/Approximate Recursive Matching

Table 5.11 gives the performance comparison of our method with the top-performing teams in the BioNLP shared task [1]. The official evaluation shows that our best results would rank 6th in extracting biological events in the testing data compared to the reported results of the 24 participating teams of the shared task. Specifically, the performance of Binding event type and the overall performance of regulation event types would all rank 5th. In addition, the results of our second best run with the matching criteria "Edge+POS+All" would rank 7th in overall F-score but rank 2nd in precision compared to the top 7 performing systems.

Team	Precision(%)	Recall(%)	F-score(%)
UTurku	58.48	46.73	51.95
JULIELab	47.52	45.82	46.66
ConcordU	61.59	34.98	44.62
UT+DBCLS	55.59	36.90	44.35
VIBGhent	51.55	33.41	40.54
DalhousieU	**41.77**	**33.66**	**37.28**
UTokyo	53.56	28.13	36.88
UNSW	45.78	28.22	34.92

Table 5.11: Performance comparison of our method with participating teams

5.4.5 Error Classification

Since the gold event annotation of testing data is hidden, we decided to examine the event extraction results of the development data based on its provided gold events to analyze the underlying errors in the results. Since the matching feature "Edge+Relaxed_POS+Trigger_Stemming" achieves the best F-score of 37.28% on the testing set, we therefore focused on the results of the development set using "Edge+Relaxed_POS+Trigger_Stemming" and analyzed 20 abstracts randomly extracted from the set. The detailed analysis is reported in terms of false negatives, the events that our method is not able to identify, and false positives, the events that our method extracts incorrectly.

False negatives

It is shown that false negative events have a substantial impact on the performance of all 24 participating teams of the shared task [1]. The best recall, 46.73%, achieved by University of Turku [46], captures less than half of the gold events in the testing set. In our work, four major causes of false negatives are determined: low coverage of event trigger tokens, low coverage of rule set, compound error effect, and anaphora and coreference.

(1) Low coverage of event trigger tokens

According to the Table 5.8, by adding only one extra matching criterion that the same event trigger tokens of the rules have to also appear in the matching sentences, "Edge+POS+Trigger" outperforms "Edge+POS" by more than 36% in F-score. This indicates that event trigger tokens play an important role in identifying biological events.

We found that many trigger tokens in the development sentences do not appear as triggers in the training set. Therefore, these trigger tokens are not covered by our event rules induced only from the training sentences. This leads to the failure of extracting the corresponding events since some trigger tokens in the sentences can not find their exact match in the event rules and therefore the matching criteria can not be satisfied. This problem contributes about one third of all false negative events.

Stemming the trigger tokens by Porter's algorithm [115] helps to detect more events by allowing the inflectional or derivational forms of the triggers to match. However, some variants of trigger tokens will still not be covered. For instance, a false negative event occurs because the trigger of a rule "coexpressing" is not able to match the hyphenated trigger "co-expressing" in the sentence as the standard stemming can not determine that both are derived from a same root word. Similar false negative results involve event triggers with other prefixes such as "down-regulation" and "up-regulation". Also, synonyms of trigger tokens can not be handled by stemming. For instance, "critical", the trigger of a rule, can not match "crucial", the trigger token in the sentence, although they are synonyms and every other token of the rule can find its match in the sentence, thus leading to another false negative.

Therefore, since the training data is the only source of event trigger tokens in our work, the coverage of triggers is limited, thus limiting the generalization power of event rules when fixing trigger tokens in the matching. It has been demonstrated by UTurku, the best-performing team in the primary task of the BioNLP shared task, that a sophisticated supervised machine learning technique can better detect event triggers [46]. In their work, a multi-class support vector machine classifier was used to assign event types to individual tokens, one at a time. A wide array of features was adopted including token features, frequency features and dependency features. In addition, trigger detection parameters were tuned experimentally using a grid search to find the global optimal performance on the training set. High performance in detecting triggers in the sentences explains in part their success in the event extraction.

Using external biological lexical resources to enrich the set of event trigger tokens has also been shown effective by JULIE Lab who achieved second place in the event extraction task [116]. They manually built an event trigger dictionary using the original GENIA event corpus [44]. Therefore, the dictionary naturally contains the underlying triggers of the testing set. Then, the dictionary was assessed and further extended by two biologists based on their domain knowledge. Next, a separate event trigger disambiguation process was proposed to assign each trigger token to one fixed event type based on a conditional probability. Compared to their "once a trigger, always a trigger" method, however, triggers are treated in a more flexible way in our work. A token is not necessarily always a trigger unless it appears in the appropriate

context. Also, the same token can serve as trigger for different event types as long as it appears in the different context. A trigger will only be classified into a fixed event type when it could serve as trigger for different event types in the same context.

(2) Low coverage of rule set

Other than the low coverage of event trigger tokens, we found that many gold events are described in different grammatical structures that are not covered by the existing event rules induced from the training sentences. These structures tend to be more complex, involving a long dependency path from the event trigger to event arguments in the graph representation of sentences. Events that consist of these structures are not recognized as no matched rules will be returned from the subgraph matching. This is supported by Table 5.9 in which by matching only the dependency relations between tokens, the least specific criterion "Edge" captures only 52.17% of the gold events in the testing set. This problem results in another third of all false negatives, but could be alleviated if a bigger training set, which contains more complex events, was available.

Careful evaluation reveals three specific scenarios. First, the existing rules in the rule set cannot extract any argument of the gold event, not to mention the entire event. Second, the existing rules are not able to detect all the arguments of multi-argument events. For instance, only one theme of a two-Theme Binding event can be identified by the Binding rule set. The entire gold event is thus treated undiscovered under the primary evaluation measure of the shared task, "Approximate Span Matching/Approximate Recursive Matching". Third, an existing rule is capable of extracting the entire gold event if some modifications of dependency relation can be made to the rule without changing the meaning of the relation. For example, the dependency representation $prep\_of(increase, immunoreactivity)$ in rules should match with $prep\_in(increase, immunoreactivity)$ in sentences in order to successfully extract the gold event as "increase of immunoreactivity" possesses a same meaning as "increase in immunoreactivity". In addition, the most generic relation, dependent (dep), should be relaxed to match any relation, so $dobj(increase, BIO\_Entity)$ in rules can match $dep(increase, BIO\_Entity)$ in sentences. However, a significant amount of manual work is required to carefully tune the grammatical relations of the

Stanford dependency representation [91].

(3) Compound error effect

Defined in the shared task, events of Regulation, Positive_regulation and Negative_regulation types can take sub-events as arguments. Therefore, if the nested sub-events are not correctly identified, the main events will not be extracted due to the compound error effect which contributes another source of false negatives.

(4) Anaphora and coreference

Since our proposed graph matching-based method focuses on extracting biological events from sentences, events that contain protein names that span multiple sentences will not be captured. Recognition of these events from the data requires the ability to do anaphora and coreference resolution in biological text [117–119]. This problem, together with the compound error effect, accounts for the rest one third of false negative effects.

False positives

Four major causes of false positives are generalized from our analysis: loose matching criteria, assignment of overlapped event rules, lack of postprocessing rules, and inconsistencies in gold annotation. Each of the problems contributes about one fourth of all false positive events.

(1) Loose matching criteria

Some false positives are generated due to the looseness of the matching feature "Edge+Relaxed_POS+Trigger_Stemming" in two aspects. First, it allows trigger tokens to match if the stemmed tokens are identical. Some tokens are thus erroneously matched because they happen to have the same root word. For instance, the trigger token "activation" of a Positive_regulation rule is able to match with the token "activity" of a sentence because they share a same root word "activ" after stemming. Then, "activity" is treated as the event trigger, resulting in an erroneous

Positive_regulation event for the sentence. Second, the matching feature allows non-trigger tokens to match as long as they possess a same POS tag. This, however, leads to a number of wrong events. For example, "NF-kappa B-dependent BIO_Entity gene" is able to match a Regulation rule originally induced from a phrase "Kappa B-dependent BIO_Entity promoter" in a training sentence as they have a same event trigger "B-dependent" and every other token has its match. However, "gene" is semantically distinct from "promoter" although they appear in the same context and share a same relaxed POS tag "NN", leading to a wrong Regulation event. Similarly, erroneous Gene_expression events are also found caused by the POS tag matching between non-trigger tokens "expression" and "gene". More stringent matching criteria between tokens are expected to help with this problem.

(2) Assignment of overlapped event rules

In our work, when rules of different event types are overlapped with each other, we keep the rule of the event type in which the trigger occurs most frequent as a trigger in the training data, and remove rules of other event types. This method, while simplistic, effectively improves the event extraction performance by greatly reducing the number of event candidates. However, it also leads to errors. One such example is: "levels" are used as the trigger for two overlapped rules of event types Transcription and Positive_regulation. Since "levels" is used more often as a Transcription trigger in the training data, the overlapped Positive_regulation rule is then removed. Therefore, an erroneous Transcription event is then detected from a development sentence instead of the gold Positive_regulation event. Although the trigger and the argument are all identified correctly, the event type is assigned wrongly.

A more sophisticated approach based on the conditional probability of an event trigger given an event type from training data, as applied in the work of JULIE Lab [116], could aid in better assigning overlapped event rules to the correct event type.

(3) Lack of postprocessing rules

The postprocessing rules applied in our work to screen candidate events come directly from the specifications of the BioNLP shared task. They can help with the general filtering purpose, however, some misidentified events require customized postprocessing rules. For instance, a Gene_expression event is detected from the phrase "Tax expression vector" of a development sentence because an expression event is annotated from a standalone phrase"Tax expression" in the training set. However, since "Tax expression" is only used as an adjective to describe the token "vector" in this context, the identified Gene_expression event is not appropriate. Likewise, "Sp1 transcription" should not be identified as an event in the context of "Sp1 transcription factors". In addition, two Gene_expression events are detected from "BIO_Entity DNA binding subunit" using "binding" and "binding subunit" as trigger respectively as the phrase can match two event rules. However, the event triggered by "binding" should not be recognized according to the gold annotation since a more specific trigger token is always preferred. These problems can be addressed effectively by adding corresponding postprocessing rules.

(4) Inconsistencies in gold annotation

Some extracted events are considered biologically meaningful but evaluated as false positives due to the inconsistencies in the gold annotation. Some such examples are provided below: (a) a Binding event is annotated from "p49(100) DNA binding subunit" in training data but not annotated for "p50 DNA binding subunit" in the development set. (b) a Positive_regulation event and a Gene_expression event are annotated with a same trigger "overexpression" from "Overexpression of BIO_Entity is implicated." in training data. However, only one Gene_expression event triggered by "overexpression" is annotated for "BIO_Entity overexpression is required." in the development set, and the Positive_regulation event is missing. (3) a Binding event is annotated from "hRXR alpha receptor" in training data but not annotated for "c-erbA/T3 receptor" within a same context in the development set. These are the evidence for the difficulty of annotating biological events correctly and consistently.

5.5 Summary

In this chapter, dependency graphs are used to automatically induce biological event rules from annotated events. These rules are then used to extract biological events from the biological literature. The extraction process is treated as a subgraph matching problem to search for a subgraph isomorphic to the graph of an event rule within the graph of a sentence. We conducted the experiments on the GENIA event corpus to cope with the primary task of the BioNLP shared task. We further explored different methods of refining the obtained event rules for the purpose of improving the performance of the event extraction. By means of the primary evaluation measure of the shared task, our method achieves an 41.84% F-score on the development data and an 37.28% F-score on the testing data in detecting biological events across nine event types, ranking 6th compared to the reported results of the 24 participating teams. We consider our results important for two reasons: (1) our method does not require any manual intervention. (2) our method does not resort to any external domain-specific resources, and could thus be generalized to extract events from other domains where training data is available.

Chapter 6

Conclusion

In this thesis, we have presented three works around biological entity recognition and biological relation extraction. These two topics are closely related as the poor performance in the entity recognition will have detrimental effects on relation extraction [59]. For biological entity recognition, we extract characters of representative biological prefixes and suffixes to aid the entity recognition system to annotate biological entities. For biological relation extraction, we identify sentences that describe interactions between biological entities using patterns defined as a sequence of specialized POS tags that capture the structure of key sentences in the scientific literature. Furthermore, we detect biological events from texts using subgraph matching by matching dependency graphs of sentences with the dependency graphs of event rules that encode the detailed descriptions and the typical contextual structures of biological events.

The chapter is organized as follows: In section 6.1, we conclude the contributions of the thesis. In section 6.2, we discuss the directions for future work.

6.1 Contributions

In the first work, we presented an unsupervised method to automatically extract domain-specific prefix and suffix characters from biological corpora. The extraction method is based on the use of PATRICIA trees [65], and a probabilistic model is proposed to determine whether a substring is both an English affix and a domain-specific affix. The detailed procedure of the PATRICIA tree-based affix extraction is shown in Algorithm 1.

As one application of our proposed method, the extracted affix characters are integrated into the parametrization of the ABTA system [16]. The system achieves an overall classification accuracy of 87.5% in locating biological terms based on five word position classes, and an 0.702 F-score in exact entity matching annotation,

which are all comparable to the experimental results obtained by the original ABTA system. However, our method helps the system significantly reduce the size of feature set and thus improves the system efficiency. The system also obtains a classification accuracy of 81.46% based only on our extracted affix information. This demonstrates that the affix information achieved by the proposed method is important to accurately locating biological entities. The evaluation on the JNLPBA shared task on biological entity recognition [22] suggests that the extracted affix information is more sensitive in identifying proteins, DNA and RNA.

We further explored the reliability of our method by gradually increasing the proportion of training data from 0.25% to 90% of the GENIA corpus. One-tailed t-Test results confirm that the ABTA system integrated with our method achieves more reliable performance than the original ABTA system when the training corpus is small. The demonstration of this extraction method implies that domain-specific affixes can be identified for arbitrary corpora without the need to manually generate training sets. The extracted affix characters can be used to aid in the domain-specific entity recognition.

In the second work, we proposed an approach to automatically extract sentence patterns that contain interactions involving concepts of molecular biology. A PATRICIA tree [65] is used for the first time to facilitate the automatic extraction of biological interaction patterns. A pattern is defined in this work as a sequence of specialized Part-of-Speech (POS) tags that capture the structure of key sentences in the scientific literature. Our pattern learning method has a time complexity of $O(n)$ in the size of candidate sentences, and presented in Algorithm 2.

Each candidate sentence for the classification task is encoded as a POS array and then aligned to a collection of pre-extracted patterns using an end-space free alignment algorithm [80]. The quality of the alignment is expressed as a pairwise alignment score. The most innovative component of this work is the use of a Genetic Algorithm (GA) to maximize the classification performance of the alignment scoring scheme. Two evaluation measures are investigated in the fitness function of GA: statistical analysis and classification accuracy.

We conducted the experiments on our designed system based on the GENIA corpus. By means of a genetic algorithm to define the scoring scheme, the system achieves

on average 0.845 F-score using GENIA annotations and 0.796 F-score using our system annotations by cross-evaluating 400 sentences in which at least two biological concepts co-occur. We further investigated the impact of different verb tags on the system performance. Based on fewer interaction patterns, the system adopting new tags achieves a noticeable improvement in performance on testing sentences in terms of both GENIA and our system annotations compared to the original system.

In addition, within the framework of our system, we conducted experiments on the same dataset for sentence identification using interaction patterns generated by another state-of-the-art pattern generating algorithm (PGA) [36] in order to compare with the performance of patterns obtained by our proposed pattern learning method. The results show that patterns generated by PGA do not perform as well as patterns obtained by our method. We infer that our proposed method is more efficient in producing interaction patterns to identify interaction sentences.

In the third work, we defined biological event rules and designed a graph-based algorithm to automatically learn event rules based on the use of gold events annotated by domain experts. Biological event rules encode the detailed descriptions and the typical contextual structures of biological events. Training sentences are represented by dependency graphs and the event rules are induced by extracting the shortest dependency path connecting the event trigger to each event argument. The detailed procedure of the algorithm is presented in Algorithm 4.

As the most innovative component of this work, we explored the application of the subgraph matching problem in the BioNLP field of extracting biological relations or events. We defined subgraph isomorphism in the context of biological event extraction, and we developed a subgraph matching algorithm using a backtracking approach, namely "Injective Graph Embedding Algorithm", to extract biological events by searching for subgraphs isomorphic to the dependency graphs of event rules within the dependency graphs of sentences. The algorithm has an estimated time complexity of $O(|R| \cdot n^2 k^n)$ where n is the total number of vertices in the rule graph and the sentence graph, k is the vertex degree, and $|R|$ is the number of event rules. Since the input graphs of sentences and event rules are small, our algorithm is efficiently solving the subgraph matching problem in this work. The main and the recursive part of the algorithm are formalized in Algorithm 5 and Algorithm 6 respectively.

We conducted the experiments on the GENIA event corpus to cope with the primary task of the BioNLP shared task on event extraction [1]. We further explored three different methods of refining the automatically learned event rules for the purpose of improving the performance of the event extraction. We also investigated the impact of different combinations of graph matching features on the performance of event extraction. By means of the primary evaluation measure of the shared task, our subgraph matching-based event extraction method achieves an 41.84% F-score on the development data and an 37.28% F-score on the testing data in detecting biological events across nine event types, which would rank 6th compared to the reported results of the 24 participating teams [1]. In the end, we examined the event extraction results and reported the detailed analysis on both false negatives and false positives, which is valuable for future improvements.

The achieved performance is considered important for two reasons: (1) our event extraction method does not require any manual intervention. (2) our event extraction method does not resort to any external domain-specific resources, and could thus be generalized to extract events from other domains where training data is available.

6.2 Future Work

The following topics are of considerable research interest to us for improving biological information extraction in future work:

For biological entity recognition, since the ABTA system employs a word-based method which annotates each word without taking previously assigned tags into account, we would like to incorporate the extracted prefix and suffix characters into other biological entity recognition systems using sequence-based methods, for instance Hidden Markov Model (HMM) that takes previous annotation decisions into account in order to decide on the tag for the current word.

Specifically, two HMM-based methods have been compared in terms of the JNLPBA shared task on biological entity recognition [22]: the traditional HMM approach and the Perceptron HMM algorithm [11]. The results show that the overall F-score of the traditional HMM approach is 53.9%, which is higher than the provided baseline but lower than the ABTA system. The results of the Perceptron HMM algorithm are more promising, that is, 69.1% without resorting to external resources.

However, characters of a fixed length are still adopted as the affix feature. We plan to integrate our method into the Perceptron HMM algorithm to investigate the impact of the extracted affix information on the performance of biological entity recognition.

For biological relation extraction, we are interested in investigating the impact of more sophisticated approaches of refining overlapped event rules across event types on the biological event extraction performance. Borrowing the idea of JULIE Lab [116], we attempt to employ the approach based on the conditional probability of an event trigger given an event type from training data to aid in assigning overlapped event rules to the correct event type.

Furthermore, we will experiment with more matching criteria when mapping event rules to sentences. For instance, since we only attempted to relax the matching criteria of POS tags and trigger tokens, we want to relax edge labels by manually tuning the grammatical relations of the dependency representation based on the existing hierarchy of typed dependencies [91]. Moreover, we plan to expand the coverage of event trigger tokens using external lexical resources, such as UMLS SPECIALIST lexicon [120] and WordNet [121], for new event triggers and synonyms of existing triggers. In addition, once the specific relations or events are discovered, further reasoning processes can be performed by means of available biological ontologies, such as UMLS Semantic Network [122] and Gene Ontology [20], to infer more general relations from facts extracted from the biological literature.

Bibliography

[1] Jin-Dong Kim, Yoshinobu Kano Tomoko Ohta, Sampo Pyysalo, and Jun'ichi Tsujii. Overview of bionlp'09 shared task on event extraction. In *Proceedings of the NAACL-HLT 2009 Workshop on Natural Language Processing in Biomedicine (BioNLP'09)*, pages 1–9. ACL, 2009.

[2] MEDLINE. National Library of Medicine. Accessed in August 2010, http://www.ncbi.nlm.nih.gov/PubMed/.

[3] Duncan Hull, Steve R. Pettifer, and Douglas B. Kell. Defrosting the digital library: Bibliographic tools for the next generation web. *PLoS Computational Biology*, 4(10), 2008.

[4] Katrin Fundel, Robert Küffner, and Ralf Zimmer. Relex—relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371, 2007.

[5] Hyunchul Jang, Jaesoo Lim, Joon-Ho Lim, Soo-Jun Park, Kyu-Chul Lee, and Seon-Hee Park. Finding the evidence for protein-protein interactions from pubmed abstracts. *Bioinformatics*, 22(14):e220–e226, 2006.

[6] Muhammad Abulaish and Lipika Dey. Biological relation extraction and query answering from medline abstracts using ontology-based text mining. *Data & Knowledge Engineering*, 61(2):228–262, 2007.

[7] Sophia Ananiadou and John Mcnaught. *Text Mining for Biology And Biomedicine*. Artech House Publishers, 2005.

[8] Aaron Michael Cohen and William R. Hersh. A survey of current work in biomedical text mining. *Briefings in Bioinformatics*, 5(1):57–71, 2005.

[9] Kristofer Franzn, Gunnar Eriksson, Fredrik Olsson, Lars Asker Per Lidn, and Joakim Cster. Mining the Biomedical Literature in the Genomic Era: An Overview. *J. Comp. Biol.*, 10(6):821–855, 2003.

[10] Goran Nenadic and Sophia Ananiadou. Mining semantically related terms from biomedical literature. *ACM Transactions on Asian Language Information Processing (TALIP)*, 5(1):22–43, 2006.

[11] Sittichai Jiampojamarn, Grzegorz Kondrak, and Colin Cherry. Biomedical term recognition using discriminative training. In *Proceedings of Recent Advances in Natural Language Processing (RANLP 2007)*, Borovets, Bulgaria, 2007.

[12] GuoDong Zhou and Jian Su. Exploring deep knowledge resources in biomedical name recognition. In *the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*, 2004.

[13] Burr Settles. Biomedical named entity recognition using conditional random fields and novel feature sets. In *Joint wsh. on NLP in Biomedicine and its Applications (JNLPBA-2004)*, 2004.

[14] Dan Shen, Jie Zhang, Guodong Zhou, Jian Su, and Chew-Lim Tan. Effective adaptation of a Hidden Markov Model-based named entity recognizer for biomedical domain. In *Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine*, pages 49–56, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[15] Ki-Joong Lee, Young-Sook Hwang, and Hae-Chang Rim. Two-phase biomedical NE recognition based on SVMs. In *Proc. of the ACL 2003 workshop on Natural language processing in biomedicine*, pages 33–40, Morristown, NJ, USA, 2003. ACL.

[16] Sittichai Jiampojamarn, Nick Cercone, and Vlado Kešelj. Biological Named Entity Recognition using N-grams and Classification Methods. In *Conf. of the Pacific Assoc. for Computational Linguistics, PACLING'05*, Tokyo, Japan, 2005.

[17] Andre Skusa, Alexander Ruegg, and Jacob Kohler. Extraction of biological interaction networks from scientific literature. *Brief Bioinform*, 6(3):263–276, 2005.

[18] Jiao Li, Xian Zhang, Yu Hao, Minlie Huang, and Xiaoyan Zhu. Learning domain-specific knowledge from context–THUIR at TREC2005 Genomics Track. In *Proceedings of 14th Text Retrireval Conference (TREC2005)*, Gaithersburg, USA, 2005.

[19] Soumya Raychaudhuri. *Computational Text Analysis: For Functional Genomics and Bioinformatics*. Oxford University Press, 2006.

[20] Gene Ontology Consortium. Creating the gene ontology resource: design and implementation. *Genome Research*, 11(8):1425–1433, 2001.

[21] Jin-Dong Kim, Tomoko Ohta, Yuka Teteisi, and Jun'ichi Tsujii. Genia corpus - a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl. 1):i180–i182, 2003.

[22] Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*, 2004.

[23] Michael Krauthammer and Goran Nenadic. Term identification in the biomedical literature. *J. of Biomedical Informatics*, 37(6):512–526, 2004.

[24] K. Fukuda, T. Tsunoda, A. Tamura, and T. Takagi. Toward information extraction: Identifying protein names from biological papers. In *the Pacific Symposium on Biocomputing*, pages 707–718, 1998.

[25] Kristofer Franzén, Gunnar Eriksson, Fredrik Olsson, Lars Asker, Per Lidén, and Joakim Cöster. Protein names and how to find them. *I. J. Medical Informatics*, 67(1-3):49–61, 2002.

[26] K. Bretonnel Cohen, Philip V. Ogren, Lynne Fox, and Lawrence Hunter. Corpus design for biomedical natural language processing. In *ISMB '05: Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases*, pages 38–45, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[27] Chih Lee, Wen-Juan Hou, and Hsin-Hsi Chen. Annotating multiple types of biomedical entities: a single word classification approach. In *JNLPBA '04: Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 80–83, Morristown, NJ, USA, 2004. Association for Computational Linguistics.

[28] Yu Song, Eunju Kim, Gary Geunbae Lee, and Byong Kee Yi. POSBIOTM-NER in the shared task of BIONLP/NLPBA 2004. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*, 2004.

[29] William B. Cavnar and John M. Trenkle. N-gram-based text categorization. In *Proc. SDAIR-94, 3rd Ann. Symposium on Doc. Analysis and Inf. Retr.*, pages 161–175, Las Vegas, USA, 1994.

[30] K.Park, S.Kim, D.Lee, and H.Rim. Boosting lexical knowledge for biomedical named entity recognition. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*, 2004.

[31] Jenny Finkel, Shipra Dingare, Huy Nguyen, Malvina Nissim, Gail Sinclair, and Christopher Manning. Exploiting context for biomedical entity recognition: From syntax to the web. In *Joint wsh. on NLP in Biomedicine and its Applications (JNLPBA-2004)*, 2004.

[32] Guodong Zhou, Jie Zhang, Jian Su, Dan Shen, and Chewlim Tan. Recognizing names in biomedical texts: a machine learning approach. *Bioinformatics*, 20(7):1178–1190, 2004.

[33] Jun'ichi Kazama, Takaki Makino, Yoshihiro Ohta, and Jun'ichi Tsujii. Tuning support vector machines for biomedical named entity recognition. In *Proceedings of the ACL-02 workshop on Natural language processing in the biomedical domain*, pages 1–8, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[34] Carol Friedman, Pauline Kra, Hong Yu, Michael Krauthammer, and Andrey Rzhetsky. Genies: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17 Suppl 1, 2001.

[35] Kevin Humphreys, G. Demetriou, and R. Gaizauskas. Two applications of information extraction to biological science: Enzyme interactions and protein structures. In *Proceedings of the Pacific symposium on Biocomputing*, pages 502–513, Hawaii, USA, 2000.

[36] Minlie Huang, Xiaoyan Zhu, Yu Hao, Donald G. Payan, Kunbin Qu, and Ming Li. Discovering patterns to extract protein-protein interactions from full texts. *Bioinformatics*, 20:3604–3612, 2004.

[37] Fabio Rinaldi, Gerold Schneider, Kaarel Kaljurand, Michael Hess, Christos Andronis, Ourania Konstandi, and Andreas Persidis. Mining of relations between proteins over biomedical scientific literature using a deep-linguistic approach. *Artif. Intell. Med.*, 39(2):127–136, 2007.

[38] Haibin Liu, Christian Blouin, and Vlado Keselj. Sentence identification of biological interactions using patricia tree generated patterns and genetic algorithm optimized parameters. *Data Knowl. Eng.*, 69(1):137–152, 2010.

[39] John Wilbur, Lawrence Smith, and Lorraine Tanabe. Biocreative 2. gene mention task. In *Proceedings of Second BioCreative Challenge Evaluation Workshop*, pages 7–16, 2007.

[40] Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski1. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC Bioinformatics*, 9 Suppl 11:s2, 2008.

[41] Mary Elaine Califf and Raymond J. Mooney. Bottom-up relational learning of pattern matching rules for information extraction. *Journal of Machine Learning Research*, 4:177–210, 2003.

[42] Claire Nédellec. Machine learning applied to information extraction in specific domains - an example, gene interaction extraction from bibliography in genomics, 2002.

[43] AK Ramani, RC Bunescu, RJ Mooney, and EM Marcotte. Consolidating the set of known human protein-protein interactions in preparation for large-scale mapping of the human interactome. *Genome Biology*, 6(5), 2005.

[44] Jin-Dong Kim, Tomoko Ohta, and Jun ichi Tsujii. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9(1):10, 2008.

[45] Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Björne, Jorma Boberg, Jouni Järvinen, and Tapio Salakoski. Bioinfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8(1):50, 2007.

[46] Jari Björne, Juho Heimonen, Filip Ginter, Antti Airola, Tapio Pahikkala, and Tapio Salakoski. Extracting complex biological events with rich graph-based feature sets. In *BioNLP '09: Proceedings of the Workshop on BioNLP*, pages 10–18, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[47] Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski. A graph kernel for protein-protein interaction extraction. In *BioNLP '08: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 1–9, Morristown, NJ, USA, 2008. Association for Computational Linguistics.

[48] Alexander S. Yeh, Lynette Hirschman, and Alexander A. Morgan. Evaluation of text data mining for database curation: Lessons learned from the KDD Challenge Cup. *Bioinformatics*, 19 (Suppl. 1):331–339, 2003.

[49] Christian Blaschke, Miguel A. Andrade, Christos Ouzounis, and Alfonso Valencia. Automatic extraction of biological information from scientific text: Protein-protein interactions. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 60–67. AAAI Press, 1999.

[50] Anna Divoli and Teresa K. Attwood. BioIE: extracting informative sentences from the biomedical literature. *Bioinformatics*, 21(9):2138–2139, 2005.

[51] Temple F. Smith and Michael S. Waterman. Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197, 1981.

[52] Vlado Keselj, Haibin Liu, Norbert Zeh, Christian Blouin, and Chris Widden. Finding optimal parameters for edit distance based sequence classification is NP-hard. In *Proceedings of KDD-09 Workshop on Statistical Relational Mining and Learning in Bioinformatics (StReBio'09)*, Paris, France, 2009.

[53] Haibin Liu, Christian Blouin, and Vlado Kešelj. Identifying interaction sentences from biological literature using automatically extracted patterns. In *Proceedings of BioNLP 2009*, Boulder, Colorado, USA, 2009. NAACL/HLT 2009 Workshop.

[54] Conrad Plake, Jorg Hakenberg, and Ulf Leser. Learning patterns for information extraction from free text. In *Proceedings of AKKD 2005*, Karlsruhe, Germany, 2005.

[55] Jung-Hsien Chiang and Hsu-Chun Yu. Literature extraction of protein functions using sentence pattern mining. *IEEE Transactions on Knowledge and Data Engineering*, 17(8):1088–1098, 2005.

[56] Razvan Bunescu, Ruifang Ge, Rohit J Kate, Edward M Marcotte, Raymond J Mooney, Arun K Ramani, and Yuk W Wong. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial Intelligence in Medicine*, 33(2):139–155, 2005.

[57] Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Raymond J. Mooney, Yuk Wah Wong, Edward M. Marcotte, and Arun Ramani. Learning to extract proteins and their interactions from medline abstracts. In *Proceedings of ICML-2003 Workshop on Machine Learning in Bioinformatics*, pages 46–53, 2003.

[58] Jorg Hakenberg, Conrad Plake, Ulf Leser, Harald Kirsch, and Dietrich Rebholz-Schuhmann. Lll'05 challenge: Genic interaction extraction with alignments and finite state automata. In *Proceedings of Learning Language in Logic Workshop (LLL'05) at ICML*, page 38C45, Bonn, Germany, 2005.

[59] Renata Kabiljo, Andrew B Clegg, and Adrian J Shepherd. A realistic assessment of methods for extracting gene/protein interactions from free text. *BMC Bioinformatics*, 10:233, 2009.

[60] Rune Saetre, Kenji Sagae, and Jun'ichi Tsujii. Syntactic features for protein-protein interaction extraction. In *Short Paper Proceedings of the Second International Symposium on Languages in Biology and Medicine (LBM)*, 2007.

[61] Fabio Rinaldi, Gerold Schneider, Kaarel Kaljurand, James Dowdall, Christos Andronis, Andreas Persidis, and Ourania Konstanti. Mining relations in the GENIA corpus. In *Proceedings of the Second European Workshop on Data Mining and Text Mining for Bioinformatics*, Pisa, Italy, 2004.

[62] Kenji Sagae, Yusuke Miyao, Takuya Matsuzaki, and Jun'ichi Tsujii. Challenges in mapping of syntactic representations for framework-independent parser evaluation. In *the Workshop on Automated Syntactic Annotations for Interoperable Language Resources*, 2008.

[63] Sampo Pyysalo, Filip Ginter, Katri Haverinen, Juho Heimonen, Tapio Salakoski, and Veronika Laippala. On the unification of syntactic annotations under the stanford dependency scheme: a case study on bioinfer and genia. In *BioNLP '07: Proceedings of the Workshop on BioNLP 2007*, pages 25–32, Morristown, NJ, USA, 2007. Association for Computational Linguistics.

[64] Yusuke Miyao, Kenji Sagae, Rune Saetre, Takuya Matsuzaki, and Jun'ichi Tsujii. Evaluating contributions of natural language parsers to protein–protein interaction extraction. *Bioinformatics*, 25(3):394–400, 2009.

[65] Donald R. Morrison. Patricia - Practical Algorithm To Retrieve Information Coded in Alphanumeric. *Journal of the ACM*, 15(4):514–534, 1968.

[66] Lee-Feng Chien. Pat-tree-based keyword extraction for chinese information retrieval. *SIGIR Forum*, 31(SI):50–58, 1997.

[67] Keh-Jiann Chen, Wen Tsuei, and Lee-Feng Chien. Pat-trees with the deletion function as the learning device for linguistic patterns. In *Proceedings of the 17th international conference on Computational linguistics*, pages 244–250, Morristown, NJ, USA, 1998. Association for Computational Linguistics.

[68] The Brown Corpus. Brown University. Accessed in August 2010, http://clwww.essex.ac.uk/w3c/corpus_ling/.

[69] Ethem Alpaydin. *Introduction to Machine Learning*. MIT Press, 2004.

[70] A statistical tagger. Lingua::EN::Tagger. Accessed in August 2010, http://search.cpan.org/~acoburn.

[71] Toshihide Ono, Haretsugu Hishigaki, Akira Tanigami, and Toshihisa Takagi. Automated extraction of information on protein-protein interactions from the biological literature. *Bioinformatics*, 17(2):155–161, 2001.

[72] Martin Krallinger, Alexander Morgan, Larry Smith, Florian Leitner, Lorraine Tanabe, John Wilbur, Lynette Hirschman, and Alfonso Valencia. Evaluation of text-mining systems for biology: overview of the Second Biocreative community challenge. *Genome Biology*, 9(Suppl 2):S1, 2008.

[73] Claire Nédellec. Learning Language in Logic Genic Interaction Extraction Challenge. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 31–37, 2005.

[74] Andrei Mikheev. Periods, capitalized words, etc. *Comput. Linguist.*, 28(3):289–318, 2002.

[75] Haibin Liu, Christian Blouin, and Vlado Kešelj. An unsupervised method for extracting domain-specific affixes in biological literature. In *Proceedings of BioNLP 2007*, Prague, Czech Republic, 2007. Association for Computational Linguistics.

[76] Lance Ramshaw and Mitch Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey, 1995.

[77] J. J. Cimino and G. O. Barnett. Automatic knowledge acquisition from medline. *Methods of Information in Medicine*, 32(2):120–130, 1993.

[78] Vasileios Hatzivassiloglou and Wubin Weng. Learning anchor verbs for biological interaction patterns from published text articles. *International Journal of Medical Informatics*, pages 19–32, 2002.

[79] S.B. Needleman and C.D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453, 1970.

[80] Gusfield Dan. *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1997.

[81] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 3(52), 1965.

[82] SAS Institute Inc. SAS Procedure Guide, Version 8, Cary, NC: SAS Institute Inc., 1999.

[83] John R. Koza. *Genetic Programming*. The MIT Press, 1992.

[84] Canonical genetic algorithm. AI::Genetic. Accessed in August 2010, http://search.cpan.org/dist/AI-Genetic/.

[85] Claire Nedellec, Mohamed Ould Abdel Vetah, and Philippe Bessières. Sentence filtering for information extraction in genomics, a classification problem. In *PKDD '01: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 326–337, London, UK, 2001. Springer-Verlag.

[86] Lawrence Hunter, Zhiyong Lu, James Firby, William A. Baumgartner Jr., Helen L. Johnson, Philip V. Ogren, and K. Bretonnel Cohen. Opendmap: An open source, ontology-driven concept analysis engine, with applications to capturing knowledge regarding protein transport, protein interactions and cell-type-specific gene expression. *BMC Bioinformatics*, 9, 2008.

[87] Halil Kilicoglu and Sabine Bergler. Syntactic dependency based heuristics for biological event extraction. In *BioNLP '09: Proceedings of the Workshop on BioNLP*, pages 119–127, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[88] Yuanyuan Tian, Richard C. Mceachin, Carlos Santos, David J. States, and Jignesh M. Patel. Saga: a subgraph matching tool for biological graphs. *Bioinformatics*, 23(2):232–239, 2007.

[89] Xifeng Yan, Feida Zhu, Jiawei Han, and Philip S. Yu. Searching substructures with superimposed distance. In *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering*, page 88, Washington, DC, USA, 2006. IEEE Computer Society.

[90] Huahai He and Ambuj K. Singh. Closure-tree: An index structure for graph queries. In *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering*, page 38, Washington, DC, USA, 2006. IEEE Computer Society.

[91] Marie-Catherine de Marneffe and Christopher D. Manning. The Stanford typed dependencies representation. In *CrossParser '08: Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Morristown, NJ, USA, 2008. Association for Computational Linguistics.

[92] Official Task Website. Bionlp'09 Shared Task on Event Extraction. Accessed in August 2010, http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/SharedTask/.

[93] Sampo Pyysalo, Filip Ginter, Katri Haverinen, Juho Heimonen, Tapio Salakoski, and Veronika Laippala. On the unification of syntactic annotations under the stanford dependency scheme: a case study on bioinfer and genia. In *BioNLP '07: Proceedings of the Workshop on BioNLP 2007*, pages 25–32, Morristown, NJ, USA, 2007. Association for Computational Linguistics.

[94] Andrew B. Clegg and Adrian J. Shepherd. Benchmarking natural-language parsers for biological applications using dependency graphs. *BMC Bioinformatics*, 8:24, 2007.

[95] Cartic Ramakrishnan, Pablo N. Mendes, Shaojun Wang, and Amit P. Sheth. Unsupervised discovery of compound entities for relationship extraction. In *EKAW '08: Proceedings of the 16th international conference on Knowledge Engineering*, pages 146–155, Berlin, Heidelberg, 2008. Springer-Verlag.

[96] Endika Bengoetxea. *Inexact Graph Matching Using Estimation of Distribution Algorithms*. PhD thesis, Ecole Nationale Supérieure des Télécommunications, Paris, France, Dec 2002.

[97] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[98] J. R. Ullmann. An algorithm for subgraph isomorphism. *J. ACM*, 23(1):31–42, 1976.

[99] Shijie Zhang, Shirong Li, and Jiong Yang. Gaddi: distance index based subgraph matching in biological networks. In *EDBT '09: Proceedings of the 12th International Conference on Extending Database Technology*, pages 192–203, New York, NY, USA, 2009. ACM.

[100] David Emms, Richard C. Wilson, and Edwin R. Hancock. Graph matching using the interference of continuous-time quantum walks. *Pattern Recogn.*, 42(5):985–1002, 2009.

[101] Duck Hoon Kim, Il Dong Yun, and Sang Uk Lee. Attributed relational graph matching based on the nested assignment structure. *Pattern Recogn.*, 43(3):914–928, 2010.

[102] Luigi P. Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(10):1367–1372, 2004.

[103] Marcello Pelillo, Kaleem Siddiqi, and Steven W. Zucker. Matching hierarchical structures using association graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(11):1105–1120, 1999.

[104] Fedor V. Fomin, Daniel Lokshtanov, Venkatesh Raman, B. V. Raghavendra Rao, and Saket Saurabh. Faster algorithms for finding and counting subgraphs. *CoRR*, abs/0912.2371, 2009.

[105] Gérard P. Huet. A unification algorithm for typed lambda-calculus. *Theor. Comput. Sci.*, 1(1):27–57, 1975.

[106] Yoshinobu Kano, William A. Baumgartner, Luke McCrohon, Sophia Ananiadou, K. Bretonnel Cohen, Lawrence Hunter, and Jun'ichi Tsujii. U-compare. *Bioinformatics*, 25(15):1997–1998, 2009.

[107] Siddhartha Jonnalagadda, Luis Tari, Jörg Hakenberg, Chitta Baral, and Graciela Gonzalez. Towards effective sentence simplification for automatic processing of biomedical text. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 177–180, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[108] Yoshimasa Tsuruoka, Yuka Tateishi, Jin-Dong Kim, Tomoko Ohta, John McNaught, Sophia Ananiadou, and Jun ichi Tsujii. Developing a Robust Part-of-Speech Tagger for Biomedical Text. LNCS 3746:382–392, 2005.

[109] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423–430, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[110] Christian F. Hempelmann, Vasile Rus, Arthur C. Graesser, and Danielle S. McNamara. Evaluating state-of-the-art treebank-style parsers for coh-metrix and other learning technology environments. In *EdAppsNLP 05: Proceedings of the second workshop on Building Educational Applications Using NLP*, pages 69–76, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[111] Daniel Cer, Marie-Catherine de Marneffe, Daniel Jurafsky, and Christopher D. Manning. Parsing to stanford dependencies: Trade-offs between speed and accuracy. In *7th International Conference on Language Resources and Evaluation (LREC 2010)*, 2010.

[112] Rong Xu, Alex Morgan, Amar K. Das, and Alan Garber. Investigation of unsupervised pattern learning techniques for bootstrap construction of a medical treatment lexicon. In *BioNLP '09: Proceedings of the Workshop on BioNLP*, pages 63–70, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[113] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2001.

[114] Sriram Pemmaraju and Steven Skiena. *Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Cambridge University Press, 2003.

[115] M. F. Porter. An algorithm for suffix stripping. pages 313–316, 1997.

[116] Ekaterina Buyko, Erik Faessler, Joachim Wermter, and Udo Hahn. Event extraction from trimmed dependency graphs. In *BioNLP '09: Proceedings of the Workshop on BioNLP*, pages 19–27, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[117] Caroline Gasperin. Semi-supervised anaphora resolution in biomedical texts. In *BioNLP '06: Proceedings of the Workshop on Linking Natural Language Processing and Biology*, pages 96–103, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[118] Caroline Gasperin and Ted Briscoe. Statistical anaphora resolution in biomedical texts. In *COLING '08: Proceedings of the 22nd International Conference on Computational Linguistics*, pages 257–264, Morristown, NJ, USA, 2008. Association for Computational Linguistics.

[119] Jose Castano, Jason Zhang, and James Pustejovsky. Anaphora resolution in biomedical literature. In *Proceedings of the International Symposium on Reference Resolution for NLP*, Alicante, Spain, 2002.

[120] T. McCray, S. Srinivasan, and A. C. Browne. Lexical methods for managing variation in biomedical terminologies. In *Proc 18th Annual Symposium on Computer Applications in Medical Care*, pages 235–239, 1994.

[121] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[122] Alexa T. Mccray and Olivier Bodenreider. A conceptual framework for the biomedical domain. In *Semantics of Relationships, Kluwer*, pages 181–198. Kluwer Academic Publishers, 2002.