Relation Extraction Methods for Biomedical Literature

Bùi Quốc Chính

Relation Extraction Methods for Biomedical Literature

Bùi Quốc Chính

RELATION EXTRACTION METHODS FOR BIOMEDICAL LITERATURE

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. D.C. van den Boom
ten overstaan van een door het college voor promoties
ingestelde commissie,
in het openbaar te verdedigen in de Agnietenkapel
op donderdag 6 september 2012, te 12:00 uur

door

Bui Quoc Chinh

geboren te Thai Nguyen, Vietnam

The work described in this thesis has been carried out at the Section Computational Science, University of Amsterdam, with the financial support by the DynaNets project (http://www.dynanets.org) EU grant agreement no: 233847 and the Vietnamese Oversea Training Program (http://www.vied.vn).

Author contact: bqchinh@gmail.com

ISBN 978-90-6464-578-5

Everything should be made as simple as possible, but not one bit simpler.

Albert Einstein

To my family…

Contents

Chapter 1. Introduction

1.1 Motivation

Biomedical data is crucial for research in life sciences such as systems biology and medicine. For instance, systems biology needs data to understand properties of protein-protein interactions (PPIs) [1]. Interactions data, e.g. data stored in PPIs databases, are used to validate the reliability of large-scale experimental PPI datasets [2, 3]. Networks generated from these interactions are useful for multiple purposes, e.g. for predicting new PPIs and for finding novel associations between genes and diseases[4–6]. Furthermore, experimental data are also systematically linked with the results of different studies derived from published literature to obtain a better understanding of the biological systems, and to find interesting associations among disparate facts, leading to the discovery of new or unsuspected knowledge [7, 8]. In medicine, up-to-date drug resistant information is vital to have better treatment for many diseases [9, 10]. *In silico*, biomedical data are used to build computational models to simulate how viruses, for example human immunodeficiency virus (HIV), interact with the human immune system in order to find an optimal drug regiment for individual treatment [11–13]. HIV epidemic data is used to study how these viruses are spread due to social interactions in order to have a better prevention strategy [14]. Above all, to facilitate these studies, data need to be in structured form for easy access and for obtaining evidence [15–18]. A typical example that shows the role of biomedical data in identifying diseases is illustrated in Figure 1.1.

Vast amount of biomedical data are available in unstructured form through scientific publications. Together with the development of high-throughput experimental techniques and computational models, this amount of data is being generated with an exponential rate [7, 10, 17]. Traditional search engine such as Google or specialized information retrieval tools such as PubMed provide modest help. With a few keywords, PubMed or Google can return thousands of relevance documents, but the users still need to read all those returned documents to find the data they need. Although new requirements for publishing biomedical results have been proposed such as nanopublication [19] or annotated digital abstracts [20], it is becoming more and more difficult to discover knowledge or generate scientific hypotheses without the use of data extraction techniques [16, 17]. At the same time, the number of biological databases and their entries, e.g. BIND [21], BioGrid [22], and HPRD [23], grow steadily but most of information is added by database curators and does come directly not from the original research. To build such PPIs databases, literature curators have to read all documents related to the field. This is a time consuming and laborious task. Furthermore, a recent study by [24] shows that the quality of literature-curated PPI databases is relatively low and can only cover a small fraction of data actually discovered. One of the reasons is that new discovered data are published in a wide range of scientific journals which spread into many

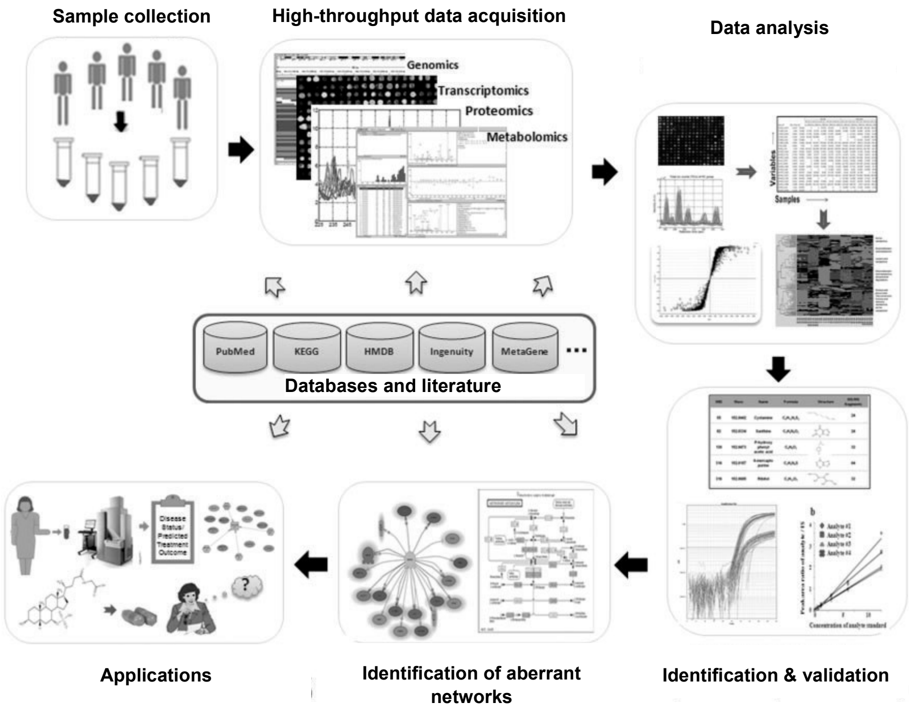


Figure 1.1 The central role of biomedical data in life sciences (adapted from [16])

disciplines and have a weak link to each other. Therefore many PPIs are overlooked by the curators. For these reasons, there is increasing interest in techniques that can automatically extract relations between entities such as PPIs from biomedical text and present the distilled knowledge to users in concise and structured form [25]. Studying these relation extraction methods is the main focus of this thesis.

There are many challenges to extract relations from biomedical text which requires new and different methods compared to the existing approaches being used for general text. First, the high ambiguity of vocabulary, long and complex sentences in biomedical text cause performance of natural language processing (NLP) tools, which trained on general English corpora, to drop considerably [26, 27]. Therefore the performance of many relation extraction methods that work well for newswire text degrades significantly when applied to text in the biomedical domain. Second, the lack of standard gene and protein names and their synonyms make the recognition of name entities (NER) mentioned in biomedical text, which is a prerequisite step of relation extraction, a difficult task [3, 28]. Furthermore, the high degree of variation in biomedical terminologies also contributes to this problem, which then degrades the overall performance of the extraction systems [25]. Finally, the availability of high quality annotated corpora is scarce since they are expensive and time-consuming to produce. Such corpora are important to train NLP tools as well as machine learning (ML) algorithms for extracting relations in biomedical text [29–32]. Due to these challenges, extracting relations from biomedical text has been an active research field during the last decade.

Although many approaches have been proposed, extracting relations from biomedical text remains a big issue due to, among others, the quality of the extracted relations, performance time (speed), and the type of relations being extracted [28, 33–37]. First, the performance of extraction systems, which is measured in terms of precision, needs to be improved to satisfy the demand of aforementioned tasks such as building high quality biological databases. Second, most of the proposed systems require a significant performance time when applied for large scale extraction. Therefore, these systems are not ready for real time application. Third, existing approaches have mainly focused on extracting PPIs, and recently on *biomedical events*; many relation types are still untouched.

1.2 Research questions

The aim of this thesis is to study methods for relation extraction from biomedical text. In particular, we focus on extracting three types of relations, namely causal relations on HIV drug resistance, protein-protein interactions, and *biomedical events*. Furthermore, these relation extraction methods are investigated under three different scenarios that are commonly encountered in this research field: no training data is available, training data is available but relation types are missing, and full training data are available, respectively. The research questions that we address in this thesis are as follows:

1. How is syntactic information used for different relation extraction tasks? Syntactic information has been used in most relation extraction systems. The use of syntactic information depends on the levels of NLP analysis ranging from part-of-speech (POS) to deep parsing. In this thesis we study the use of syntactic information for three relation extraction methods in accordance with the scenarios above. This research question is addressed in Chapter 3, 4 and 5.
2. What is the role of machine learning to relation extraction task? ML methods play an important role in relation extraction systems. The use of ML methods depends on the availability of training data as well as the properties of data. In this study we demonstrate that ML methods can effectively be used to leverage the performance of existing relation extraction tasks. The role of machine learning to relations extraction task is discussed in Chapter 4 and 5.
3. Which factors contribute to the performance time of a relation extraction system? When the system is applied to large-scale extractions then

computational resources required to train and run the system should be taken into account. There are many factors contribute to the performance time of a relation extraction system such as which NLP tools are used to analyze input text, how many features are used for the ML classifier. Here we show that by applying a data partition strategy for input texts, the performance time of the ML-based systems can be boosted significantly. This research question is addressed in Chapter 4 and 5.

The detailed answers to each research question are given in chapter 6.

1.3 Outline of the thesis

Following the aim of relation extraction: "to distil knowledge and present it in a concise form to users" [6], this thesis is written in a concise form. When details are needed, readers are directed to the specific references. This thesis consists of the following chapters:

Chapter 2 provides background for relation extraction methods in biomedical text. We start with an overview of a typical relation extraction system such as its work-flow and which tools and techniques are commonly used. We then discuss the role of NLP and ML tools for relation extraction tasks. Finally we characterize main techniques and present their state-of-the-art results on relation extraction tasks.

Chapter 3 presents a novel method to extract causal relations on HIV drug resistant based on grammatical rules. We show how these rules are formulated and how to combine the extracted relations. First, we review existing methods to extract binary relations. We then present our method and discuss its results and some possible ways to improve the performance of our method.

Chapter 4 introduces a new method to extract PPIs from biomedical text. We start with an introduction to the existing PPIs extraction approaches. We then describe our method and show how to use a ML classifier as a filter to relation extraction task. Finally, we discuss the evaluation results on five PPI corpora and compare our results with the results of the other systems.

Chapter 5 focuses on methods to extract *biomedical events* from text. We begin with an overview of the *biological events* and existing methods used to extract these events. We then introduce our approach to automatically learn rules from training data and how to apply these rules to new text.

Chapter 6 answers the research questions raised in Chapter 1 and summarizes this thesis with overall discussion and conclusion.

Chapter 2. Relation extraction methods for biomedical text

Abstract

This chapter provides a background and a review of existing techniques for extracting relations from biomedical text. We start with an overview of a relation extraction system. We then focus on each component employed in the extraction system, namely name entity recognition (NER), NLP tools, and ML methods. For each component being discussed, we provide a list of popular tools that are available for use. Furthermore, we also discuss the role of annotated corpora, how they influence the selection extraction methods. Finally, we present main approaches and their state-of-the-art results of relation extraction from biomedical text.

2.1 Introduction

With the huge amount of information hidden in biomedical text, which is measured by the numbers of publications, that is increasing with an exponential rate, it is no longer possible for a researcher to keep up-to-date with all developments in a specific field [3, 6]. In addition, the focus of biomedical research shifts from individual genes or proteins to entire biological systems, making the demand for extracting relationships between biological entities (e.g. protein-protein interactions, genes-diseases) from biomedical text to discover knowledge and to generate scientific hypotheses increasingly urgent [7, 8]. Manually transforming this information from unstructured text into a structured form is a time-consuming and laborious task [24]. Automatic relation extraction offers an interesting solution to this problem because it can reduce time spent by researchers on reviewing the literature and by significantly covering many more scientific articles than those normally reviewed [38]. Therefore, many approaches have been proposed to extract relations from biomedical text in recent years, ranging from simple co-occurrence approaches to sophisticated machine learning-based approaches. These approaches differ from each other in many respects such as how the natural language processing (NLP) techniques are used to analyze the input text and which methods are used to learn extraction rules i.e. manually defined pattern or automatically learn from training datasets [28]. In general, a relation extraction system consists of three modules, namely text preprocessing, parsing, and relation extraction as shown in Figure 2.1.

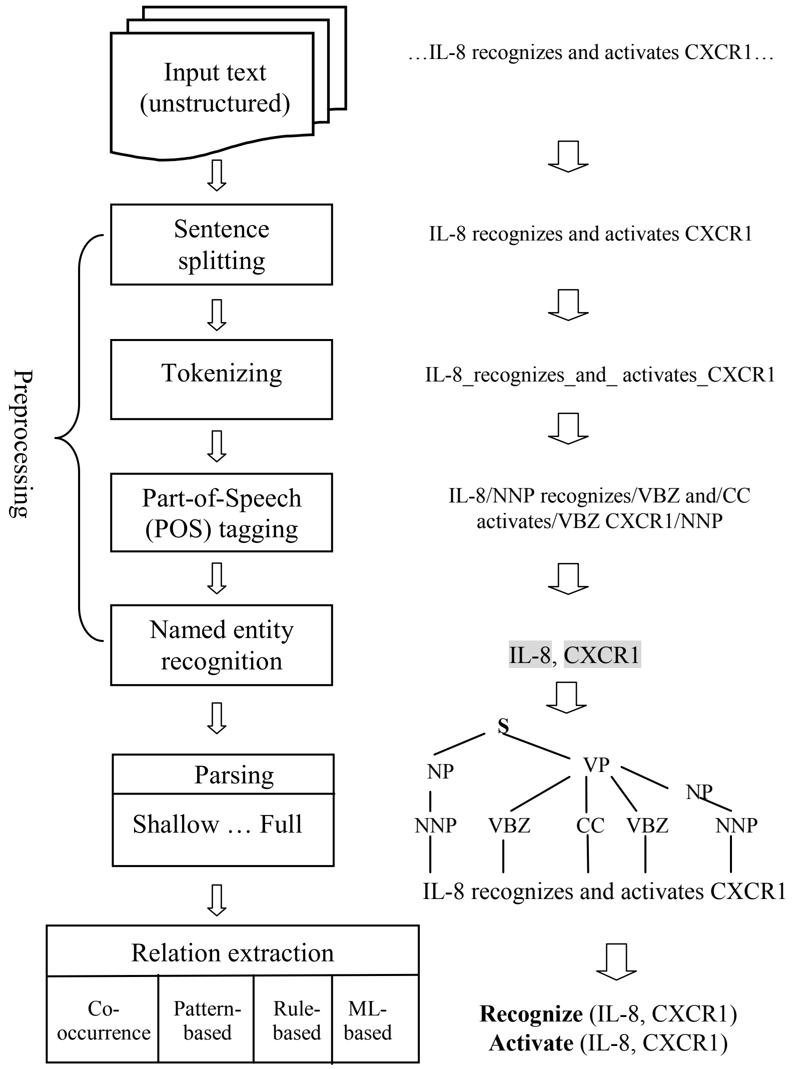

Input text (unstructured)
…IL-8 recognizes and activates CXCR1…
Preprocessing
Sentence splitting
IL-8 recognizes and activates CXCR1
Tokenizing
IL-8_recognizes_and_ activates_CXCR1
Part-of-Speech (POS) tagging
IL-8/NNP recognizes/VBZ and/CC activates/VBZ CXCR1/NNP
Named entity recognition
IL-8, CXCR1
Parsing
Shallow … Full
S
NP
VP
NP
NNP
VBZ
CC
VBZ
NNP
IL-8 recognizes and activates CXCR1
Relation extraction
Co-occurrence
Pattern-based
Rule-based
ML-based
Recognize (IL-8, CXCR1)
Activate (IL-8, CXCR1)

Figure 2.1. Workflow of a typical relation extraction system

2.1.1 Text preprocessing

Text preprocessing module splits documents into basic text units for subsequent processing. Currently, most of relation extraction systems work with sentence-based unit therefore larger blocks of text such as abstracts, paragraphs or whole documents are split into single sentences. Based on this unit, a tokenization step is carried out to further split the sentence into tokens, which are then used as input for subsequent processes such as the named entity recognition step. The sentence splitting, tokenizing, and part-of-speech tagging steps are described in more details in the NLP section.

2.1.2 Named entity recognition

The aim of biomedical named entity recognition (NER) is to identify the biomedical entities (e.g. names of genes and proteins) mentioned in text. NER is a prerequisite step for any relation extraction systems [2]. Due to the complexities of the biomedical terminology, recognizing named entities in biomedical text is considered as a daunting task [39]. The main issue in NER is the lack of standardization of names. First, each biomedical entity such as gene or protein might have several names and abbreviations, for example, gene *BRCA1* is also known as *breast cancer 1*, or *Brca1*, *BRCA 1*, and *brca1*, etc. Second, the same name can refer to different entities depending on the context e.g. *Cdc2* can be two completely unrelated genes. Third, biomedical entities may have multi-word names which cause overlap of candidate names. A recent study by [34] has shown that the performance of the same PPI extraction system, which is measured in terms of F-score, can drop 15% when applied to two identical datasets, in which, one dataset is given the gold standard protein names and the other uses a NER tool to recognize protein names.

To recognize these names, the early NER methods rely on manually defined rules which based on the characteristics of names such as morphological and syntactic features. As annotated corpora are now available, more NER systems are based on machine learning methods [40, 41]. To reduce the number of false positives, some systems also rely on dictionaries which consist of a list of synonyms of entities [42]. Currently, the performance of the state-of-the-art NER systems for biomedical texts in terms of F-scores varies from 78% to 90%. A list of available NER tools is shown in Table 2.1.

Table 2.1. List of common NER tools for biomedical text

Name	Description	URL
BANNER	Trained on BioCreative 2 GM data.	http://banner.sourceforge.net
Lingpipe	Trained on GENIA corpus.	http://alias-i.com/lingpipe/
GENIA Tagger	Trained on GENIA corpus.	http://www.nactem.ac.uk/tsujii/GENIA/tagger/
ACELA	Trained on GENIA.	http://www.nactem.ac.uk/acela/

2.1.3 Parsing

The goal of the parsing step is to transform unstructured text into more structured forms, which may reveal linguistic patterns or common similarities of the written text. Based on the parsing output, methods are built to extract relations between biomedical entities. Depending on the extraction techniques, the use of parsing tools to analyze text vary from shallow parsing to dependency parsing and deep parsing. The details of these parsing techniques are described in section 2.2

2.1.4 Relation extraction

Relation extraction is the core module of a relation extraction system. Many techniques have been proposed to extract relation from biomedical text. Early systems focus on extracting a small set of simple relations such as inhibit or binding relations between proteins while recent systems attempt to extract more complex relations such as protein-protein interactions or biomedical events. In general, techniques used in relation extraction systems can be divided into four groups, namely co-occurrence, pattern-based, rule-based, and machine learning-based approaches. The detail of each approach is described in section 2.5.

2.1.5 Evaluation metrics

To evaluate the performance of a relation extraction system, the following metrics are used: recall, precision, and the F-score.

Recall = TP/(TP + FN)

Precision = TP/(TP + FP)

F-score = 2 * Recall * Precision/(Recall + Precision),

where TP, FN, and FP are defined as:

TP (true positives): is the number of relations that were correctly extracted from input documents.

FN (false negatives): is the number of relations that the system failed to extract from input documents.

FP (false positives): is the number of relations that were incorrectly extracted from input documents.

The F-score is the harmonic mean of recall and precision.

The remaining structure of this chapter is as follows. In section 2.2, we introduce common NLP techniques used in relation extraction methods. We discuss the use of machine learning methods for relation extraction in section 2.3. We then introduce the biomedical corpora and evaluation metrics in section 2.4. In section 2.5 we present the current approaches to relation extraction.

2.2 Natural Language Processing

Natural language processing (NLP) is the process of analyzing text in natural language aiming to convert unstructured text into a structured form. Because of its complexity, the analysis of the natural language is carried out in many steps, which are: lexical, syntactic, and semantic analyses [25]. Lexical analysis is the process of converting a sentence into a sequence of tokens. Once tokens are determined, morphological analysis is carried out to map lexical variants of a word into its canonical base form. Syntactic analysis consists of the assignment of part-of-speech (POS) tags (.e.g., noun, verb, adjective, etc.) to the sequence of tokens that makes up a sentence and determining the structure of a sentence through parsing tools. Semantic analysis determines the meaning of a sentence and anaphora resolution. In this thesis we only focus on NLP techniques that are frequently used in relation extraction systems which are lexical and syntactic analyses. Currently, NLP techniques can only handle input text at the sentence level therefore sentence splitting is usually the first step in a NLP pipeline.

2.2.1 Sentence splitting

Sentence splitting is the process of determining sentence boundaries. It splits input text such as abstracts and paragraphs into single sentences and is a prerequisite step for subsequent processes [43]. Although it seems to be straightforward, sentence splitting is a non-trivial task. There are many issues that cause this task to become difficult, such as the irregularity in proteins or genes names (e.g. E. coli, p53), inline citations (e.g., Proc. Acad. Sci. 2006), and abbreviations (e.g. Dr., et al.). To improve the accuracy of the splitting methods over biomedical text, most of sentence splitting tools are necessary to re-train on biomedical corpora. Available tools such as Ling-Pipe can achieve an accuracy of 98% when evaluating on biomedical test sets.

2.2.2 Lexical processing

This process deals with how input text is split into words (tokens) and how *base* forms of a word are found.

Tokenization

Tokenization is the process of splitting the input sentence into a sequence of tokens. This is a prerequisite step before any linguistic analysis can be carried out. Errors that occur in this step propagate to the other levels and thus deteriorate the performance of subsequent tasks such as NER and relation extraction [44]. The simplest way to tokenize input text is based on white spaces and punctuation symbols. Similar to the sentence splitting process, the tokenization process encounters many problems such as abbreviation, in which words are not always separated from other tokens by white space. This abbreviation problem may also interfere with the sentence splitting process above. Another problem is the use of hyphenation since it is ambiguous to determine whether to return one or more tokens for hyphenated words. For example, Cdc2-depedent should be treated as two tokens, whereas DNA-binding should return as a single token. Moreover, in the biomedical domain, the tokenization process has also to deal with additional challenges due to domain-specific terminology, nonstandard punctuation, and orthographic patterns e.g., 12.0 +/- 1.6%, and *alpha-galactosyl-1,4-beta-galactosyl-1,4-glucosyl*. Currently, tokenization tools for biomedical text achieve an accuracy of 95%. A list of available tokenization tools is shown in Table 2.2.

Stemming

Stemming (morphological analysis) is the process of mapping various syntactic forms of a word into its canonical base form. The purpose of this process is to reduce the variation among tokens, which reduce the sparseness for lexical representation of the text [45]. For example, one can represent all morphological variants of relations between two entities A and B such as: activation of A and B, A activates B, A activated B, and A activating B, by a single relation activat(A,B).

Stemming is a standard technique which is commonly used to create the bags-of-words (BOW) features in many relation extraction systems. This tool is usually bundled with the NLP packages.

2.2.3 Syntactic processing

Syntactic processing is an important part of a NLP pipeline as this process reveals structural relationship between groups of words at the sentence level. This process consists of three stages: part-of-speech tagging, chunking and full parsing.

POS tagging

POS tagging is the first step of syntactic analysis and is essential to cope with the various lexico-syntactic ambiguity forms of words. For example, the word *result* can be either a common noun or a verb, depending on its syntactic context. The POS tagger receives input as a sequence of words of a sentence, and assigns POS tags to the words of that sentence. It identifies the grammatical form of a word based on the word itself and its surrounding context. Such grammatical forms output from the POS tagger are nouns, verbs, preposition, etc., and are categorized into 8 basic groups.

Recently, most of POS taggers are built based on ML methods [46]. These methods require training data which are words with manually assigned POS tags. POS taggers which are trained on wire news corpora such as the Wall Street Journal corpus can achieve an accuracy of 98% on general texts. However, they need to be re-trained on biomedical corpora in order to achieve the same accuracy level when applied to biomedical texts [47]. A list of available POS taggers which are commonly used in relation extraction system is shown in Table 2.2.

In relation extraction systems, output of a POS tagger are used to form patterns for pattern-based systems or used as features for ML-based systems. An example of POS tags is shown in Figure 2.1. The use of POS tags is discussed in section 2.5

Shallow parsing

Shallow parsing is the process of combining sequences of words into syntactic groups such as noun and verb phrases using both lexical and POS sequence information. It is a preliminary stage to help fully parsing a sentence. In shallow parsing, the structure of a sentence is not resolved to the level of single elements. Similar to POS tagging, most of modern shallow parsers are relied on the availability of training corpora which are annotated with chunks. To achieve high accuracy on biomedical text, retraining these shallow parsers on annotated biomedical corpora is essential. Currently, the performance of the state-of-the- art shallow parsers achieve an accuracy of 96% when evaluated on biomedical test corpora [27]. A list of available shallow parser is shown in Table 2.2. An output of shallow parsing is shown in Figure 2.2a.

Many relation extraction systems employ shallow parsing to analyze input text. These systems range from rule-based to ML-based approach. The detailed usage of the shallow parsing output is discussed in section 2.5.

Full parsing

Full parsing is the process of analyzing syntactic structure of a sentence with the most elaborated details. It accumulates the output of all previous steps such as POS tags, phrases (chunks) and adds more information about structural dependencies between phrases. The full parsing output of a given sentence is a parse tree which reveals the relationship of subject-predicate-object of that sentence. Full parsing techniques can be divided into three types: phrase structure parsing, dependency parsing, and deep parsing [26]. Phrase structure parsing produces a constituent tree of a sentence, where syntactic tags are inner nodes and words are leaves, as shown in Fig 3. Dependency parsing produces tree structure of a sentence, where nodes are words and edges represent the relationships among words, as shown in Figure 2.4. Deep parsing produces theory-specific syntactic and semantic structures and predicate argument structures which represents the relationship among words [48]. With the availability of biomedical treebanks, existing general purpose parsers which are trained on general text such as the WSJ corpus are now able to retrain on biomedical domain. The performance of the state-of-the-art full parser for biomedical text achieves an accuracy of 80%. A list of available full parser is shown in Table 2. Examples of full parsing are shown in Figure 2.1 and Figure 2.2.

a) [NP IL-8] [VP recognizes and activates] [NP CXCR1]

b)

c)

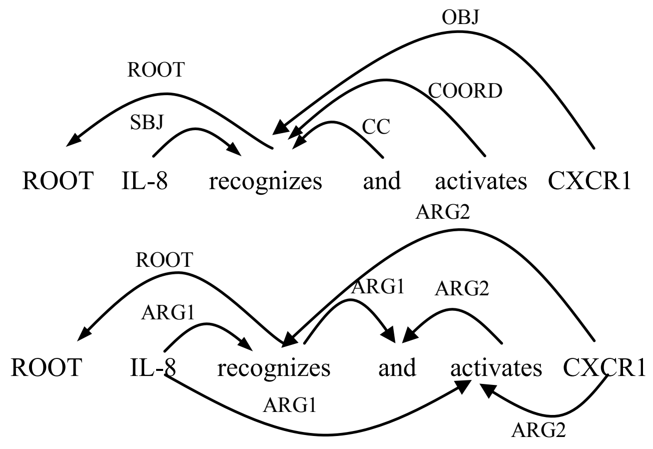

OBJ
ROOT
COORD
SBJ
CC
ROOT IL-8 recognizes and activates CXCR1
ARG2
ROOT
ARG1
ARG2
ARG1
ROOT IL-8 recognizes and activates CXCR1
ARG1
ARG2

Figure 2.2. Parsers output of sentence "IL-8 recognizes and activates CXCR1". (a) output of the shallow parser. (b) output of the dependency parser. (c) output of the deep parser in form of predicate argument structure.

The use of full parsing in relation extraction systems vary depending on the parsing types as well as extraction techniques. The detailed usage of the full parsing output is discussed in section 2.5.

2.3 Machine learning

Machine learning (ML) techniques are widely used as a component of relation extraction methods. In this section we introduce the basic concepts of the ML techniques and explain how they are used in the relation extraction systems. The details of general ML algorithms and specific algorithms for relation extraction can be found in [49] and [33], respectively.

Machine learning methods are based on statistical analysis of data to infer general rules. It stems from the situation that we do not explicitly know solutions to solve the problems but data related to these problems are available. In relation extraction, such data can be annotated text, sentences, or phases containing relations between entities. The task of a ML method is either to learn rules from these examples which reveal the structure of the underlying data, or to distinguish instances of data from each other. Therefore, the outcome of a ML method is either the learning rules or a model which is used to predict unknown data based on previous seen data [49]. For example, given a set of biomedical data containing gene-disease associations, a ML method then learns a model to predict gene-disease associations from unseen biomedical data. In this case, the learning method corresponds to the supervised machine learning paradigm, which consists of two phases: training and testing phase. In the training phase, ML methods build a model by learning sets of properties and their respective values of the examples from the training data. Such properties are called features. It is based on the observation that examples belongs to the same class (e.g. interaction, non-interaction) have more features in common than examples that belong to other classes. Such features are then used by ML methods to decide whether an example belongs to a specific class or not. Therefore, most ML algorithms try to find a set of features, their values, and combinations of features to distinguish all classes from each other. Although many ML methods have been proposed, the most commonly used ML methods for relation extraction are support vector machines (SVM) [50].

Table 2.2. A list of common NLP tools for biomedical text.

Category	Name	URL
Sentence splitter	Lingpipe	http://alias-i.com/lingpipe
	Enju	http://www.nactem.ac.uk/y-matsu/geniass
Tokenization	OpenNLP	http://opennlp.apache.org
	Stanford	http://nlp.stanford.edu/software/tokenizer.shtml
	Lingpipe	http://alias-i.com/lingpipe
POS tagger	Enju	http://www.nactem.ac.uk/GENIA/tagger/
	OpenNLP	http://opennlp.apache.org/
	Stanford	http://nlp.stanford.edu/software/tokenizer.shtml
	Lingpipe	http://alias-i.com/lingpipe/
Shallow parser	OpenNLP	http://opennlp.apache.org/
	Illinois Chunker	http://cogcomp.cs.illinois.edu/page/software
	Lingpipe	http://alias-i.com/lingpipe/
Full parser	Stanford	http://nlp.stanford.edu/software/lex-parser.shtml
	Charniak–Lease re-ranking	ftp://ftp.cs.brown.edu/pub/nlparser/reranking-parserAug06.tar.gz
	OpenNLP	http://opennlp.apache.org/
	Enju	http://www.nactem.ac.uk/enju/

2.3.1 SVM methods for relation extraction

Most relation extraction systems use ML methods as the classifiers. In a nutshell, these methods work as follows. Given a training set, for example, a list of annotated sentences, some of which contain PPI pairs (positive examples) and some contain non-interacting protein pairs (negative examples). All sentences are transformed into representations such that ML methods can capture properties or features that are best expressed the interaction pairs (or not for negative examples). The simplest representation form is a list of words that occur in the sentences. More complex representations are parse trees obtained from the output of NLP tools which can reveal the structure and dependencies of words in the sentence. The set of structured representations and the PPIs are then used as input for a ML classifier i.e. SVM classifier to learn a model of how PPIs are typically expressed. To predict new PPI pairs from unseen text, every new sentence must be transformed into the same representation as the training sentences, the SVM classifier then use the learned model to classify whether each candidate PPI pair is an interaction or not. The mechanism for a SVM classifier to carry out such classification is to find a hyperplane that can separate positive and negative examples with the largest possible margin [36, 37]. Such a hyperplane is learned from training examples. Training examples that lie closest to the hyperplane are the support vectors. To classify a new object, the SVM classifier checks on which side of the hyperplane that object resides. Imagination that these examples can be linearly separated, e.g. in two dimensional case, then we can draw a line that separates two classes as shown in Figure 2.3. A list of features that commonly used in relation extraction is described in section 2.5.

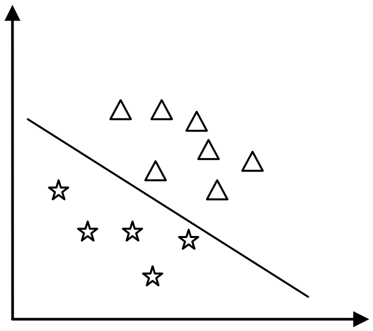


Figure 2.3. A hyperplane which separates the class of instances represented by stars from the class of instances represented by triangles.

2.3.2 Kernel methods

There are many cases where the classes in data sets are not linearly separable. In these cases, training data can be mapped into higher dimensional space through a non-linear mapping. In the new space, the classes may be separable and the hyperplane classifier can be applied. This non-linear mapping is done via a kernel function which takes the representation of two examples and computes their similarity [51].

Many kernels have been proposed, ranging from general purpose to user-defined kernels. While general kernels are applicable to arbitrary problems, user-defined kernels incorporate domain knowledge to fit specific applications. In relation extraction, user-defined kernels mainly differ from each other by their structural representations and how the similarity functions are calculated [33]. Some common kernels are discussed in section 2.5

2.4 Biomedical corpora

Biomedical corpora play an important role in the development of relation extraction methods, such as providing data to retrain existing NLP tools to work with biomedical texts, to train ML-based relation extraction approaches, and to facilitate automatic performance evaluation of extraction systems. Since creating biomedical corpora is a time-consuming and error prone task, the number of available biomedical corpora is small. A full list of the available biomedical corpora can be found in the WBI corpus repository: http://corpora.informatik.hu-berlin.de. Some typical corpora are listed bellows:

2.4.1 GENIA corpus

GENIA is the largest annotated corpus publicly available in the biomedical domain [52]. It consists of 2,000 Medline abstracts with more than 400,000 words and almost 100,000 annotations for biological terms. In addition, it contains annotation for linguistic structures such as part-of-speech, phrasal and syntactic structures. This corpus is often used to retrain NLP tools such as tokenizers, POS taggers, and full parsers to work with biomedical text.

2.4.2 PPI corpora

There is a small set of five PPI corpora, namely AIMed [53], BioInfer [54], HPRD50 [55], IEPA [56], and LLL [57]. These corpora are created independently for different purposes. Although these corpora contain annotation for named entities and PPI pairs, they vary in many aspects including size (e.g. number of abstracts ranging from 50 to 200), biological coverage (i.e. retrieved from PubMed using different keywords) and annotation policy (e.g. interaction type, direction of interactions). To minimize the differences between these corpora, [29] have transformed these PPI corpora into an XML-based format and proposed to use only undirected and untyped PPI pairs from these corpora for evaluation purposes. With the introduction of these unified PPI corpora, it is easier to compare performance between PPI extraction methods. Furthermore, new evaluation methods based on these PPI corpora have also been proposed which are cross-learning and cross-corpora [58]. In cross-learning, four corpora are used for training and the fifth corpus is used for testing, whereas in cross-corpora, one corpus is used for training and the other four are used for testing. These evaluations reveal the ability of an extraction method adapting to new text with unknown characteristics. This is one step closer to the real world situation where the input text is diversity. These PPI corpora are used to evaluate our method to extract PPI in Chapter 4.

2.4.3 GENIA events corpus

The GENIA *events* corpus is based on the GENIA corpus, consisting of 1,000 Medline abstracts [59]. It contains 9,372 sentences in which 36,114 events are identified. An event consists of a trigger, type and participants of the event. The event trigger is a word or phrase in the sentence which indicates the occurrence of the event. The event type categorizes the type of information expressed by the event. The event participants are entities or other events. Table 2.3 shows a list of defined events and their arguments. This corpus and 15 annotated full-text documents are used to evaluate our method to extract biological events in Chapter 5.

Table 2.3. Event types and their arguments for Genia event extraction task [60]. Secondary arguments are optional. P denotes for Protein, Ev denotes for event

Type	Primary Argument	Secondary Argument
Gene_expression	Theme(Protein)	
Transcription	Theme(Protein)	
Protein_catabolism	Theme(Protein)	
Phosphorylation	Theme(Protein)	Site
Localization	Theme(Protein)	AtLoc,ToLoc
Binding	Theme(Protein)+	Site+
Regulation	Theme(P/Ev),Cause(P/Ev)	Site, CauseSite
Positive_regulation	Theme(P/Ev),Cause(P/Ev)	Site, CauseSite
Negative_regulation	Theme(P/Ev),Cause(P/Ev)	Site,CauseSite

2.5 Relation extraction methods

Various approaches have been proposed to extract relations from biomedical text [58, 61–66]. Due to the inherent complexity of biomedical text, most of relation extraction systems work on sentence-based level. The approaches used in relation extraction systems vary from the level of linguistic analysis to the way patterns or rules are being learned. Based on the techniques employed in these systems, we can categorize them into four groups, namely co-occurrence, pattern-based, rule-based, and ML-based approaches. In the following sections we present the most common characteristics of these methods (while ignoring their targeted relation types). The detailed methods to extract the specific type of relations such as PPIs and biological events are left in the related chapters.

2.5.1 Co-occurrence approaches

Co-occurrence is the simplest approach to identify relationship between two entities that co-occur in the same sentence, abstract, or document [5]. This approach is based on the hypothesis that if two entities are frequently mentioned together, it is likely that they are somehow related. Since two entities might be mentioned together without any relation, most systems use frequency-based scoring schemes to eliminate those relations occurred by chance [8]. The more unlikely the observed relation, the stronger the relation between entities is scored by the system. Co-occurrence approaches tend to achieve high recall but suffer low precision since biomedical texts usually consist of complex sentences which contain multiple entities but only small fraction of these entities are actually related or have relationships. For example, in the AIMed corpus, only 17% of protein pairs that belong to the same sentence actually describe protein-protein interactions [29]. However, the precision of these systems can be improved by applying filtering steps e.g. aggregation of single PPI at corpus level, removal of sentences that do not match lexico-syntactic criteria, or requiring the occurrence of a relation word between two candidate proteins [34]. Another drawback of these methods is that the types of relationships and their direction are not known. Therefore these approaches are not suitable for applications that require fine-grained extracted relations such as annotated PPI pairs for PPI databases.

Since co-occurrence approaches are simple and do not require any linguistic analysis, they are robust in terms of performance time and are still valuable for some applications. For example, co-occurrence approach is often used as baseline method against which other methods are compared when there is no benchmark available [29]. For discovering purposes, e.g. finding relations between genes and diseases, many types of networks are created by applying co-occurrence approaches to a large amount of text, for example, using all of the PubMed abstracts. Such networks do to not provide precise relationships between entities, but they helps organizing the literature in a way that makes exploration much easier. Examples of recent work based on co-occurrence approaches are [15, 67, 68].

2.5.2 Pattern-based approaches

Pattern-based systems rely on a set of patterns to extract relations. These approaches can be categorized into two groups: manually defined patterns and automatically generated patterns.

Manually defined patterns

Systems based on manually defined patterns require domain experts involve in defining patterns, which is a time-consuming process [36]. These systems differ from each other depending on the level of linguistic analysis employed to define patterns. Earlier systems are based on word forms which usually express patterns using regular expressions, such as *Pro1 * relation * Pro2*, in which *Pro1* and *Pro2* refer to protein names while *relation* refers to the verb which describes the relationship between two proteins [69]. These systems are aimed to extract a limited set of relations given a list of predefined relation words such as *inhibit*, *bind*, *activate*, etc. Obviously, these systems are too simple to achieve satisfactory results [33]. Later systems attempt to define patterns using syntactic analysis of a sentence, such as POS tags, and phrasal structure (e.g. noun phrases, verb phrases, preposition phrases). Such patterns are referred to surface patterns and they do not generalize well on complex sentences [70]. Moreover, the closer examining the text, the more patterns are needed to take account of the large amount of surface grammatical variations in text. To overcome the drawback of surface patterns, some systems incorporate higher level of representation of a sentence i.e. syntactic and semantic structure such as subject-predicate-object or predicate argument structures (PAS) which requires deep analysis of input sentences by full parsing tools [71]. This incorporation leads to some improvements on performance as well as makes the patterns generalize better than those of surface patterns [28].

Overall, manually defined patterns achieve high precision but have relatively poor recall. When such patterns are applied to unseen data, they fail to extract relations that do not occur in the sampling (training) data which used to develop patterns. These approaches are not feasible in practical applications due to their limited generalization therefore they are not adapted well when applying to a new domain.

Automatic patterns generation

To increase the recall of systems based on manually defined patterns, automatically generated patterns are proposed. There are two techniques to generate patterns automatically: by using bootstrapping [72] or generating directly from corpora [73]. In general, bootstrapping techniques do not require a corpus. A small list of relations (e.g. PPI pairs) is given as an input (seeds). The first step is to find patterns that can extract seeds from text. In the second step, the obtained patterns are then applied to the texts in order to extract new relations of the same type. The initial list of relations is expanded to include newly obtained relations. This process is repeated to find more patterns until no more patterns can be found. To control the way patterns are generated, a distant supervised technique is applied [74, 75].The advantage of the bootstrapping methods is that the initial list of relations is relatively small. However, these approaches are prone to drift and a large number of noisy patterns are generated. In contrast, approaches which generate patterns directly from corpora require a larger number of annotated relations. In both techniques, the linguistic features used to generate patterns are more extensive than those of manually defined, ranging from lexical-based to POS tags, and to syntactic information.

Although the recall of systems using automatic pattern generation can improve, the precision is reduced due to noisy patterns [76, 77]. Moreover, the number of generated patterns is large, some of which are too specific to match unseen text, whereas some are too generic that overmatch any text. Therefore choosing the right patterns to apply for input text is a challenging problem. In order to make these patterns generalize well and to reduce the noisy patterns, many algorithms have been proposed to combine the generated patterns, ranging from dynamic programming to statistical scoring schemes [75, 78, 79].

Overall, automatic patterns generation approaches achieve better performance than those that are manual defined. These systems are capable to generalize well on unseen text. The drawback of these approaches is that the noisy patterns cause the reduction of the precision of these systems. Furthermore, some systems do not take into account the important aspects of extracted relations such as direction and negation of relations.

2.5.3 Rule-based approaches

Rule-based systems rely on a set of rules to extract relations [80, 81]. Similar to the pattern-based approaches, these extraction rules are obtained in two ways: manually defined and automatically generated from training data. In some rule-based systems, the differences between these systems and pattern-based systems are minor since they also use patterns or templates to express rules. Such rule-based systems extend the patterns by adding more constraints to resolve issues that are not easy to express using patterns such as checking negation of relations and determining direction of relations [77, 82, 83]. In contrast, for some rule-based systems, the differences between rule-based and pattern-based systems are clear. Instead of using regular expressions to represent the constraints, these systems rely on a set of rules, which usually express in form of a set of procedures or heuristic algorithms [48, 55, 84]. These rules are then applied to the syntactic structure of a sentence such as a dependency parse tree to extract relations.

Compared to pattern-based approaches, in general, rule-based approaches are more flexible since they are built based on rather abstract levels such as syntactic structures and use grammatical relations and semantic relations of the sentence. Therefore rule-based approaches may generalize well when applied to new domains. Furthermore, the number of rules is relatively smaller than those of pattern-based systems. However, these approaches also suffer from low recall since the defined rules can only cover obvious cases. To improve the recall of these systems, a trade-off between precision and recall can be achieved by relaxing the constraints or by learning rules automatically from training data.

2.5.4 Machine learning-based approaches

With the availability of annotated corpora on biomedical domain, approaches to relation extraction based on machine learning (ML) techniques become ubiquitous [85–90]. Most approaches use supervised learning, in which relations extraction tasks are modeled as classification problems. To build the classification models, data from annotated corpora are transformed into more structural forms by using various NLP tools. Although many ML-based methods have been proposed such as Bayesian network and maximum entropy methods, SVM methods are the most popular one. In general, the ML-based approaches can be categorized into feature-based methods and kernel-based methods.

Feature-based approaches

Feature-based systems use standard kernels, in which each data instance is represented as a feature vector $X=\{x_1, x_2, \ldots, x_n\}$ in an n-dimensional space. These systems treat the SVM classifier as a ‘black-box’ and mainly focus on defining features that potentially best represent the data characteristics. Various feature types have been proposed to use with the SVM classifiers ranging from lexical to syntactic information [63, 90–95]. The features which are specific to binary relations between two entities (e.g. protein-protein or gene-protein) in the following way:

Bag-of-words (BOW) features: This feature set consists of words that appear after, before, and between two entities. Some systems also use lemma forms of these words and their frequencies as features.

POS features: This feature set consists of the POS tags of words that appear after, before and between two entities.

Shortest path features: This feature set consists of syntactic information derived from the shortest path between two entities which are represented by two nodes of a parse tree (dependency parse tree or phrase structure parse tree).

Graph features: This feature set consists of the labels and weights of two graphs: parse structure sub-graph and a linear order sub-graph connecting two entities. The labels and weights of these two graphs are differentiated.

Beside these common features, many features have been proposed, including distance features (i.e. count the distance between two entities) and cues words (e.g. binding, interaction, activation, etc.) that describe the relations between entities.

In order to improve the performance of the classifiers, most approaches use a combination of these features. Some systems even make use of all of the features with the hope that these features can complement each other. However, when all feature types are used, the number of features increases significantly to hundreds of thousands of features, which in turn, degrades the performance of the system. Therefore, feature selection methods are applied to select the most discriminative features to use with the classifiers [96].

Kernel-based approaches

The limitation of the feature-based approaches is that it cannot make use of the structural representations of instances e.g. syntactic parse trees and dependency graphs. Therefore various kernels have been proposed to tackle this problem [33, 50, 51, 58, 97–99]. The main idea of the kernel methods is to quantify the similarity between two instances by computing the similarities of their representations. Some commonly used kernels are as follows:

Bag-of-words (BOW) kernel: This kernel uses feature vector as unordered sets of words, and calculates the similarity between two compared vectors [53].

Shallow linguistic (SL) kernel: This kernel is defined as the sum of two kernels, the global kernel and local context kernel. The global kernel uses feature set of three BOW vectors (after, before, and between two entities) and count common words in three vectors obtained from two compared sentences. The local context kernel uses surface and linguistic features generated from words on the left and right of two entities [87].

Sub tree (ST) kernel: This kernel uses the syntactic tree representations of sentences. It calculates the similarity between two input trees by counting the number of common sub-trees [51, 98].

Graph kernel: This kernel calculates the similarity between two input graphs by counting weighted shared paths of all possible paths. These paths are obtained from the dependency parse tree and linear order graph [58].

Combination of kernels: As each kernel is design to work with a specific type of input i.e. structural representations of parse trees or sequential words of sentences and it may have different strength and weakness. Therefore the combination of kernels has also been used in some approaches to relation extraction [50, 99]. This combination shows a slight increase in performance, however, the computational resource needed to train the classifiers also increases significantly [100].

2.5.5 Performance comparison of existing approaches to relation extraction

A meaningful performance comparison of existing relation extraction approaches can serve as important instruments both for perspective researchers who are interested in the relation extraction methods and for end users who want to apply automatic relation extraction techniques. However, such comparisons are not always available due to many reasons. First, these approaches target various relation types such as genes-diseases, drug-drug interactions, and protein-protein interactions. Second, many relation extraction systems evaluate their performance on different datasets which have various characteristics such as annotation policy, the ratio of positive and negative examples. Third, even if these systems are aimed to extract the same type of relation and are evaluated on the same test set, their results are not comparable due to different evaluation settings e.g. training data are split using instance-based vs. document-based level. Therefore it is not straightforward to directly compare the performance of existing systems in a fair and meaningful manner.

To facilitate a fair and straightforward performance comparison between extraction approaches, many competitive evaluations have been held. The official results from these competitions show that performances of the participant systems are far from achieving satisfaction results in order to directly integrate into real life applications. For examples, the official results of the BioCreative II.5 community challenge on extracting protein interactions show that the best system performance achieves an F-scores only above 30% [66], whereas recent evaluation of the BioNLP'11 Shared task for the GENIA event task show that the results in terms of F-scores range from 12% to 50% using strict matching criteria [60]. Furthermore, there have been some attempts to independently evaluate the existing PPI extraction methods to obtain a realistic performance on biomedical text. A study by [34] shows that performance of a typical rule-based system on the PPI extraction tasks is better than the state-of-the-art ML-based systems when evaluated using cross-corpora and cross-learning criteria. This finding is once again confirmed in a recent study by [33] on comprehensive benchmark of kernel methods to extract PPIs. In this study the authors conclude that the performance of kernel-based methods is very sensitive to parameter settings and depends largely on the corpora of which they are trained on and evaluated on. This means that extrapolating their measure performance on arbitrary text is highly problematic. Table 2.4 and Table 2.5 present performance of existing methods for PPI and biological events extraction tasks, respectively. The results show that ML-based approaches perform slightly better than the other approaches. However, it is unclear which method is the best to perform arbitrary relation extraction task. For example, the kernel-based system proposed by [45] performs better than the pattern-based system of [74] on the AIMed corpus but the pattern-based system of [74] outperforms system of [45] on the BioCreative-PPI corpus.

Table 2.4. Performance comparison of existing PPI approaches on AIMed and BioCreative-PPI (BC-PPI) corpora. Systems run on the BC-PPI corpus are evaluated at document level, whereas systems run on the AIMed corpus are evaluated at mention level.

Approach	Recall	Precision	F-score	Corpus	Ref.
Co-occurrence	51.5	22.8	31.6	AIMed	[34]
Pattern-based	71.8	48.4	57.3	AIMed	[74]
Rule-based	50.0	40.0	44.0	AIMed	[29]
Feature-based	71.9	60.0	65.2	AIMed	[63]
Kernel-based	66.6	62.7	64.2	AIMed	[45]
Kernel-based	34.5	53.1	37.4	BC-PPI	[45]
Pattern-based	43.4	43.4	42.9	BC-PPI	[74]

Table 2.5. Evaluation results of some participant teams of the BioNLP'11 Shared Task on Genia event extraction using strict matching criteria.

Team	Approach	Recall	Precision	F-Score	Ref.
UMASS	ML-based	43.89	57.96	49.95	[101]
UTurku	ML-based	45.59	53.00	49.02	[102]
MSR-NLP	ML-based	44.63	50.18	49.02	[103]
ConcordU	Rule-based	39.06	53.44	45.13	[64]
CCP-BTMG	Pattern-based	29.55	55.13	38.48	[73]

2.6 Conclusion

In this chapter we have described the components of a typical relation extraction system such as text preprocessing, named entity recognition (NER), parsing, and relation extraction. We explained the role of the text preprocessing step, the challenges of the NER task in biomedical texts and how it affects the performance of a relation extraction system. We have discussed the natural language processing (NLP) steps used to convert unstructured text into structured representations and explained how these can be used for relation extraction tasks. Furthermore, we discussed the application of machine learning (ML) methods with a focus on support vector machines for the relation extraction tasks and the biomedical corpora required by these methods. Finally, we briefly presented four main approaches to the relation extraction tasks, namely co-occurrence, pattern-based, rule-based, and ML-based approaches and discussed their performances on typical tasks. The details for specific relation extraction task are being discussed in subsequent chapters.

Chapter 3. Extracting causal relations on HIV drug resistance from literature

Abstract

In HIV treatment it is critical to have up-to-date resistance data of applicable drugs since HIV has a very high rate of mutation. These data are made available through scientific publications and must be extracted manually by experts in order to be used by virologists and medical doctors. Therefore there is an urgent need for a tool that partially automates this process and is able to retrieve relations between drugs and virus mutations from literature.

In this chapter we present a novel method to extract and combine relationships between HIV drugs and mutations in viral genomes. Our extraction method is based on natural language processing (NLP) which produces grammatical relations and applies a set of rules to these relations. We applied our method to a relevant set of PubMed abstracts and obtained 2,434 extracted relations with an estimated performance of 84% for F-score. We then combined the extracted relations using logistic regression to generate resistance values for each <drug, mutation> pair. The results of this relation combination show more than 85% agreement with the Stanford HIVDB for the ten most frequently occurring mutations. The system is used in 5 hospitals from the Virolab project (www.virolab.org) to preselect the most relevant novel resistance data from literature and present those to virologists and medical doctors for further evaluation.

[*] This chapter is based on Bui et al., “Extracting causal relations on HIV drug resistance from literature”. *BMC Bioinformatics*, **11**:101 (2010)

3.1 Introduction

The Human immunodeficiency virus (HIV) is the cause of acquired immunodeficiency syndrome (AIDS). HIV infection is now recognized as a pandemic. As of January 2006 the World Health Organization estimate that AIDS has killed over 25 million people since it was first recognized in 1981[104]. Treatment of HIV infection consists of highly active antiretroviral therapy (HAART), a multi-drug treatment and has been shown to be effective in suppressing viral replication in many patients. However, the long-term use of these drugs leads to drug resistance caused by the viral mutations that occur under drug pressure. The resulting treatment failure requires new treatment regimens that can suppress the new mutations [105]. Therefore, in HIV treatment, it is critical to have up-to-date drug resistance data for selecting a treatment regimen to which the virus is still susceptible in the presence of resistant mutations.

To assist physicians in selecting the most suitable treatment regimen, currently there are two methods available to predict HIV drug resistance: a rule-based approach [106] and recently a computational approach [107, 108]. For the former systems such as Stanford HIVDB (http://hivdb.stanford.edu) and RegaDB (http://www.rega.kuleuven.be/cev/regadb), HIV drug resistance data are updated with resistance data manually gleaned from scientific publications by experts in this field. However, the amount of biomedical literature regarding to HIV drug resistance is increasing rapidly and it is becoming highly labor intensive for experts to collect reliable drug resistance information in a convenient and effective manner. Thus, a significant amount of drug resistance data remains hidden in biomedical literature. Therefore there is a need for computational methods that automate parts of this process and that can assist in retrieving and updating causal relations between drugs and virus mutations from literature.

Several approaches for extracting relations of interest (e.g. protein-protein, gene-protein) in biomedical texts have been reported [109]. The approaches range from co-occurrence to natural language processing (NLP) techniques. Co-occurrence is the simplest approach for relation extraction of entities within sentences. It assumes that if two entities are repeatedly mentioned together, they are somehow related. This approach provides high recall (measuring the 'coverage') but very low precision (measuring the accuracy) [38].

Other approaches use pattern-based techniques to extract relations that increase precision, unfortunately at the cost of significantly lower recall [110]. The patterns are either manually defined or automatically learned through annotated data. Manual patterns are generated by domain experts through the analysis of entities connected by a specific relation from text. Automatic patterns are generated by learning from text surrounding entity pairs known to have the relationship of interest. However, the more detailed the analysis of the text, the more patterns must be taken into account to deal with the large amount of surface grammatical variation in the texts [6].

Systems that are based on NLP techniques use either shallow parsing, which divides the sentence into chunks [61, 82] or full parsing, which provides complete syntactic analysis of sentence structures. Since full parsing produces more elaborate syntactic information than shallow parsing, relation extraction systems based on full parsing can potentially provide better results [8]. The output of the parser is represented as constituent parse trees or dependency parse trees. Based on syntactic patterns or the shortest path between entities in the dependency trees, two approaches can then be applied to extract relations from parse trees: either a rule set which is manually defined [48, 55, 80] or machine learning techniques (e.g. SVM) are used [83, 88, 89].

Recent relation extraction methods focus on extraction of protein-protein interactions or protein-gene interactions [51, 92, 111]; a limited number of methods also deal with contradiction of extracted relations by assigning a strength score based on the amount of contradiction [43, 112]. Much less attention has been paid to the extraction of other types of relationships and combination of extracted relations: this research area still remains largely untouched [28].

In this paper, we introduce a novel method to extract and combine relationships between mutations in viral genomes and HIV drugs, hereafter referred to as causal relations, which express changes in the resistance to the HIV drugs which are attributed to the presence or absence of certain mutations on the HIV genome. Our system distinguishes itself from previous research on relation extraction in a number of ways. First, we apply rules to extract relations from grammatical relations of sentence constituents. Next, we combine extracted relations to generate a unique resistance value for each <drug, mutation> pair. To the best of our knowledge, this is the first attempt to apply automatic relation discovery in the field of HIV drug-ranking.

3.2 System and methods

The work-flow of the proposed method is shown in Figure 3.1. The system consists of the following components, namely text retrieval, text preprocessing, and relation extraction.

The text retrieval component collects relevant abstracts from PubMed and filters out irrelevant sentences. The text preprocessing component then simplifies sentences, parses them using the Stanford Lexicalized Parser version 1.6 [113] and applies grammatical relations to generate sentence components. The relation extraction component applies a set of rules to sentence components to extract candidate relations. Finally, the extracted relations are combined using a logistic regression classifier.

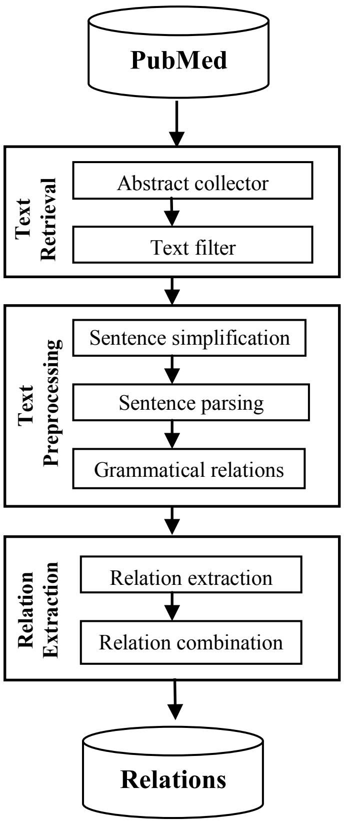


Figure 3.1. Workflow of the causal relation extraction system

3.2.1 Text retrieval

The text retrieval phase consists of two steps: collection of abstracts and selection of candidate sentences from those abstracts. To collect relevant abstracts, we prepare a list of drug names by collecting them from websites related to HIV treatment such as the Stanford HIVDB and RegaDB. The system queries PubMed using the drug names as keywords. The obtained abstracts in XML format are parsed using the LingPipe parser (http://alias-i.com/lingpipe) and then stored in a local database. Next, abstracts are split into sentences and the system selects candidate sentences that belong to either one of the following cases:

- A single sentence: if a sentence contains at least one mutation and one drug then it is selected.

- An inter-sentence: if two sentences are adjacent, one sentence contains at least one drug, and the other contains at least one mutation, then these sentences are selected.

In order to identify mutations in text, the system uses regular expressions. The regular expression for a mutation consists of an optional single letter code for the amino acid followed by a position consisting of one to three digits and ending with an amino-acid code letter [114]. Examples are K65R, I84V, and 103N. Groups of amino acids which can appear as mutations at a single position are notated with the separators “/” or “-”, such as 54A/M/V.

3.2.2 Text preprocessing

The text preprocessing phase consists of three steps: simplification of sentences, parsing the simplified sentences, and generating grammatical relations.

Simplifying sentences

Generic English parsers tend to perform poorly when applied directly to biomedical texts [3]. This is because the sentences in abstracts for such texts frequently use long and complex noun phrases and contain technical terms which are specific to the biomedical domain. For these reasons we simplify the sentences in a number of ways to make them more amenable to the parser. This process has been proposed in previous work by [80]. We further enhance this process by grouping mutations and drugs. The simplification process consists of 5 steps:

- ❖ *Removing parenthetical remarks* Words inside a pair of parenthesis () are removed except those that contain drug names or mutations.
- ❖ *Replacing "known" terms* Common terms such as "human immunodeficiency virus type 1 (HIV-1)" are replaced by their well-established abbreviations.
- ❖ *Grouping mutation and drug names* The drug names and or mutations in sentences are replaced by a predefined name. In case there is an enumerated list of drug names/mutations (either conjunctive or disjunctive), the system also replaces this group by a new name. For each sentence, the system maintains a list of generated words with the original words as a reference to be used in the extraction phase.
- ❖ *Normalizing sentences* Special characters, such as "-", "+" or "/" between words, may cause parse errors and are therefore removed.
- ❖ *Anaphora resolution* A simple anaphora resolution algorithm is implemented to resolve a list of predefined pronouns such as *this drug*, *these drugs*, etc., which refer to drug names or mutations in the sentence.

The following example illustrates the result of this simplification process:

- Original sentence: '*A371V and Q509L increased resistance to lamivudine and abacavir, but not stavudine or didanosine*'.
- Simplified sentence: '*MUTATION0 increased resistance to DRUG0, but not DRUG1*'.

Recognized keywords: In addition to mutations and drug names, the system also recognizes *relation words* and *manner words*. We prepared a list of relation words that indicate causal relations between drugs and mutations by manually analyzing sample sentences. This list is shown in Table 3.1. Furthermore, during this process we also collected adjectives and adverbs that describe the '*manner*' of the relation such as *high, strong, full, low, weak*, etc., as shown in Table 3.2. For each sentence, a list of these keywords is maintained and used in the extraction phase.

Table 3.1. Examples of relation words and their categories

Resistant	**Susceptible**	**Associated**	**Responsive**
Resistance, resistant, antagonize	Susceptibility, susceptible, sensitivity	Associate, association, bind, incorporation	Response, responsible

Table 3.2. Examples of manner words and their corresponding groups

High	Increase	Medium	Decrease	Low	No manner
High, full strong significant	Increase higher	Intermediate medium, moderate	Decrease, reduce, lower diminished	Low, weak loss	

Parsing sentences and generating grammatical relations

Before parsing, each simplified sentence is checked for a triplet <mutation, relation, drug>, in which mutation and drug are predefined names resulting from the simplification process. Sentences containing the required triplet are parsed. The parser generates the output in the form of the Penn Treebank. Figure 3.2 shows an example of the Stanford parser output. The input for the parser is the simplified sentence of the previous step.

The parse trees are then subjected to a set of English grammatical relation rules which is bundled with the parser to generate sentence constituents such as subject, object, preposition etc., which are then used as the input of relation extraction phase.

The built-in rule set consists of 49 rules, however, we only apply 11 rules that generate the most common relations, and this is shown in Table 3.3.

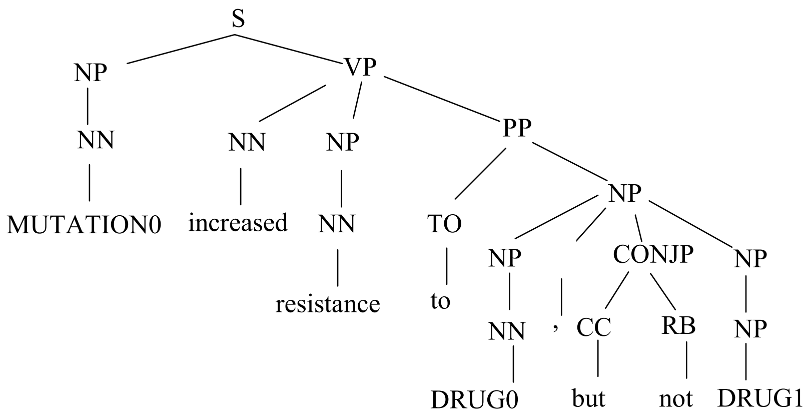


Figure 3.2. Penn Treebank output of the Stanford parser

Table 3.3. Main grammatical relations and some of their values generated from parse tree in Figure 3.2

Component	Explanation and example
nsubj	Nominal subject : MUTATION0
nsubjpass	Nominal passive subject
Pre	Predicate of a clause: increased resistance to DRUG0, but not DRUG1
dobj	Direct object: resistance
iobj	Indirect object
pobj	Prepositional object: DRUG0, but not DRUG1
prep	Prepositional modifier: to DRUG0, but not DRUG1
Cc	Coordination: but not
conj	Conjunction: DRUG1
Neg	Negation: not
acomp	Adjectival complement

3.2.3 Relation extraction

Most of the relations in biomedical texts in the English language can be expressed in two main forms:

- *Clause form*: a relation between entities is expressed by a relational verb in the form of subject and predicate (A - relation - B).
- *Phrase form*: a relation between entities is expressed by a relational noun and makes use of prepositions to connect entities (Relation - A - B).

Based on these relation forms, we define two rules which resulted from the analysis of a set of sample sentences.

Rule 1a*:* This rule applies to relations in the following form:

Subject (*keyword1*) + Predicate (*Relation word* + *keyword2*)

This is the most common relation form found in texts. If *keyword1* is MUTATION then *keyword2* is DRUG and vice versa. The procedure to extract relation of ***rule 1a*** is carried out as follows:

- *Input*: lists of sorted relation words, manner words, predefined keywords and components that belong to the predicate of the current clause.
- *Requirement*: The nominal subject (nsubj) must contain a predefined keyword (MUTATION/ DRUG).

Step 1: Find a relation pair:

a. Pick a relation word from the sorted list of relation words.

b. Find a keyword from the sorted list of predefined keywords at distance 1 to 4 words from the relation word. This keyword either belongs to the same component as the relation word or belongs to an adjacent component. If found, go to step 2, otherwise pick another relation word from the list.

Step 2: Find manner words:

a. Find a manner word from the sorted list of manner words at distance 1 to 3 words from the relation word, this manner word either belongs to the same component as the relation word or belongs to an adjacent component.

b. Continue to find other manner words at distance 1 to 2 words from this manner word.

Step 3: Extract a relation:

a. Form a pattern to extract a relation with the data found in step 1 and step 2.

b. If the list of relation words is not empty then go to step 1.

Step 4: Extract relations from a conjunction component (conj) that only contains a predefined keyword:

a. If a relation pair is found adjacent to this conj component, then use the relation word that is closest to the keyword of this component to form a relation pair.

b. Find a manner word in a similar approach as step 2.

c. Form a pattern to extract the relation from this component.

Note: In case there is more than one predefined word in *nsubj*, the first keyword is selected then the procedure will repeat for the other keywords.

Rule 1b: The same as rule 1a, but switch the role of subject and predicate for passive sentences.

Rule 1c: Preposition (*keyword1*) + Predicate (*Relation word* + *key-word2*)

This rule is similar to rule 1a, but instead of looking for a predefined keyword in *nsubj*, the system finds a predefined keyword in the preposition component (prep) that is located before the predicate.

Rule 2: This rule is applied to relations in a phrase form. First, the system calculates distance (measured by word) and numbers of occurrence of each of the following pairs in the current phrase: <Mutation, Relation>, <Drug, Relation>, <Relation, Mutation>, <Relation, Drug>. Based on these values, a heuristic algorithm forms a triple <relation, keyword1, keyword2>. Secondly, searching for *manner words* is done in the same way as described for rule 1a.

Check for negation: We can classify the negation into two cases: the negation words located outside and inside a relation. We only focus on the case where negation words are located inside a relation (see Figure 3.3). The other case is ignored since its frequency is very low and requires extensive semantic analysis. A more comprehensive analysis of negation can be found in [115]. Checking for negation is done the same way as checking for manner words.

Depending on the sentence components generated for each sentence, the system then decides to apply rule 1a, 1b or 1c, when these rules fail to extract relations, rule 2 is applied. For example, when applying rule 1a to the sentence components of the sentence in the previous step, the system forms the relation pairs as shown in Figure 3.3. The extracted relations are as follows:

MUTATION0 *increased resistance* to DRUG0

MUTATION0 ***not*** *increased resistance* to DRUG1

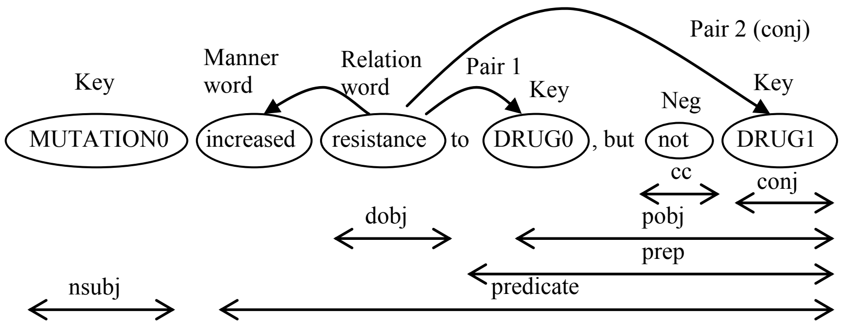


Figure 3.3. Extracting relations from grammatical relations of a simplified sentence

Post relation extraction The keywords in the extracted relations are then replaced with the original mutations or drug names. Next this group of drug names and mutations are disentangled and split into single values. For example, with the extracted relation: 'MUTATION0 increased resistance to DRUG0'. The relation after replacing keywords is as follows:

A317V and *Q509L increased resistance* to *3TC* and *ABC*

3.2.4 Relation combination

The extracted relations obtained in the relation extraction step are expressed in different manners and may also contain contradictory relations. In addition, these relations are usually taken out of context so they do not represent the true nature of the relation as it was specified in original sentences. Our task here is to determine a resistance type for each <mutation, drug> pair from these pieces of evidence, which can have the following properties:

- *Containing false positive relations due to relation extraction method or relations are taken out of context.*
- *Relations are in textual descriptive form with different manners, and come from different sources.*
- *Extracted relations contradict with each other's (resistant vs. susceptible), this is the most common case.*

Table 3.4 shows examples of extracted relations between K65R mutation and D4T. The relation combination process is carried out in two steps: grouping relations with the same mutations and drugs, and calculating the resistance type for each <drug, mutation> pair.

Grouping relations: First, the mutations in each relation are checked for consistency. Mutations with amino acid letters and those without amino acid letters are converted to a standardized form. Mutations ending with more than one amino acid letter are split into an atomic mutation, for instance M184I/V is converted into two atomic mutations, M184V and M184I. Second, extracted relations that have the same drug and mutation are put into the same group. In each group, the relations are categorized into 4 subgroups according to their resistance properties: resistant, susceptible, responsive, and associated. In addition, negative relations are removed from each subgroup. The categorizing process uses a list of predefined *relation words* some of which are shown in Table 3.1.

Table 3.4. Output extracted relations between K65R mutation and D4T when running the system over all candidate sentences

Mutation	**Relation**	**Count**	**Drug**
K65R	resistance to	5	D4T
K65R	reducing resistance to	1	D4T
K65R	result to	1	D4T
K65R	increased susceptibility to	1	D4T
K65R	fully susceptible to	2	D4T
K65R	resulted in reduced susceptibilities to	1	D4T

Calculating resistance types: Since the relations in the association and response subgroups do not indicate clear evidence on drug resistance, we only use relations from the resistance and susceptible subgroups to calculate a resistance value. For each subgroup, we divide relations into six subsets based on their *manner words* that indicate the degree of the relation. Example of the *manner words* is shown in Table 3.2. The result of this division leads to 12 subsets of relations as illustrated in Figure 3.4. Since the resistance value for common <drug, mutation> pairs are available in expert systems such as Stanford HIVDB or RegaDB, we transform the current problem into a well-known regression problem [116, 117] that is to predict the resistance value for each <drug, mutation> pair. We use the output for each <drug, mutation> pair from Stanford HIVDB as the gold standard, and use the extracted relations in each subset as features. Now the problem is to find optimized weight factor for each subset in the following equation:

$$E(y) = \sum_{i=1}^{6} (w_i r_i + w_{i+6} s_i)$$

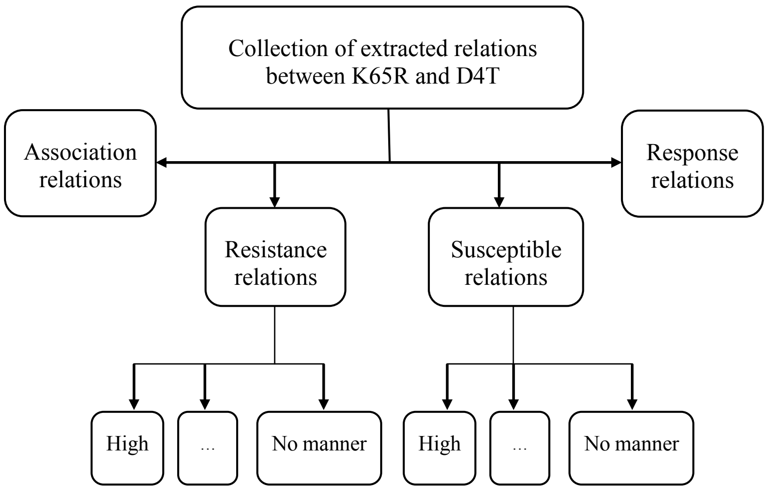


Figure 3.4. Example of categorizing extracted relations of the K65R mutation and D4T

where E(y) is the predicted value of the <mutation, drug> pair y; r_i and s_i are the number of relations in each subset of the resistance and susceptible subgroup {high, increased, medium, decreased, low, no manner} and w_j (j = 1,...,12) denote the corresponding weights of these sets.

The procedure to determine the weight factors as follows:

- Divide the extracted relations into two datasets, one for determining the weight factors (learning) and one for testing. Extracted relations for each <drug, mutation> pair can belong to either datasets. This is to make sure that learning data and test data do not overlap thus avoiding bias the final results.

- Only select a <drug, mutation> pair for training if it has at least 3 extracted relations. Assign the resistance value (resistance/ susceptible) for this pair by taking this value from Stanford HIVDB.

- Use the logistic regression function from WEKA package version 3.6 [49] to find the weigh factors.

When the weight factors in equation 1 are obtained, we then apply these values to predict the resistance value for the remaining data. Again, these predicted values are compared with the output from HIVDB system.

3.3 Results and Discussion

3.3.1 Datasets

We use the text retrieval component with a list of 22 FDA approved drugs to collect candidate abstracts from the PubMed database. We obtained 129,448 unique abstracts when the search was carried out on June 10, 2009. Among these collected abstracts there were 74,321 abstracts containing at least one drug name in the body text, 9,651 abstracts contained mutations, and 5,615 abstracts contained both drugs and mutations. When applying the text filter to find candidate sentences, we obtained 2,937 candidate sentences which contained both drugs and mutations. Of these 1,913 were single sentences and 1,024 were inter-sentences that contained the triple <mutation, relation, drug>.

Table 3.5. Datasets statistics

Dataset	Number of instances	
	Positive	Negative
500 sentences from PubMed abstracts	1095	921
130 sentences from Stanford HIVDB comments	307	261

3.3.2 Relation extraction performance

In order to evaluate the performance of the relation extraction method, we prepared two datasets: one dataset consist of sentences taken from PubMed abstracts and the other consists of sentences taken from the Stanford HIVDB comments which derived from full text papers. Table 3.5 gives an overview of the number of positive and negative relations in two datasets.

Evaluation on PubMed dataset

For dataset from PubMed, there were 1543 out of the 2937 candidate sentences containing a triple <mutation, relation word, drug>. From these, we randomly selected 500 sentences (test dataset), none of which had been used for developing rules. In this test dataset, there are 921 instances of negative relations (46%) and 1095 instances of positive relations (54%). The evaluation by experts against the output of these 500 sentences shows that the system can extract 1023 (896 true positives and 127 false positives) instances of relations with a precision, recall, and F-score of 87%, 82%, and 84.5%, respectively (see Table 3.6).

Table 3.6. Performance of the system (rule based) compared with the baseline method (Base_C) over 2 datasets. P, R, and F denote precision, recall, and F-score, respectively.

Datasets	Base_C			Rule based		
	P	R	F	P	R	F
500 sentences from PubMed abstracts	53.6	100	69.8	87.4	81.8	84.5
130 sentences from Stanford HIVDB comments	54.4	100	70	97	87	91

Evaluation of the HIVDB comments dataset

In the second evaluation, we wanted to test the performance of the proposed method on both single and inter-sentences. We used all of 130 sentences taken from the Stanford HIVDB comments, of which there are 32 sentences (24.6%) containing no relation and 98 sentences (75.4%), consisting of 56 single sentences and 42 inter-sentences. Among these 98 sentences, there are 261 instances of negative relations and 307 instances of positive relations. The evaluation shows that the system can extract 275 relations (267 true positives and 8 false positives) with a precision, recall, and F-score of 97%, 87%, and 91.7%, respectively. Table 3.6 shows the evaluation results of the system over these two datasets.

Analysis of the results

The results in Table 3.6 show that the performance of the system is comparable with existing relation extraction systems for such tasks as protein-protein interaction or protein-gene interaction, of which most do not take into account the degree of the relations [26, 28]. Furthermore, there is no existing gold standard corpus available to evaluate the results of our system, making it hard to compare the proposed method directly with other relation extraction systems. Therefore, we used a co-occurrence method as a baseline to compare with our method. This method (Base_C) predicts a <mutation, drug> pair occurring in the same sentence as a relation. Table 3.6 shows that our method has a significantly better performance than the baseline method on both datasets with F-scores of 84.5% and 91% compare to 69% and 70% of the Base_C.

The extraction results of the Stanford HIVDB dataset show a higher precision than the extraction results of datasets taken from abstracts. The reason is that the sentences taken from the Stanford HIVDB were composed in a clear, consistent way, and expressed the relations in an explicit form. In addition, the mutation and drug names are also written in a standard format and the sentences are relatively short. As a consequence, the system can archive better results. In contrast, sentences taken from abstracts are long and often have complex structures and thus are prone to more errors.

To identify the source of the errors, we analyzed 250 sentences taken from test dataset. The causes of these errors are parser errors, nonspecific rules, semantic problems, negation, and anaphora resolution:

Parser errors and grammatical relation errors: The most frequent errors were caused by the parser (23/62 i.e. 23 of the 62 failures were due to parser errors). Since the parser is not trained on biomedical texts, it often returns inaccurate parse trees, which in turn generate incorrect grammatical relations. As a result, the system applies inappropriate rules to extract relations. For instance, "*G48M causes high-level SQV resistance and intermediate resistance to NFV, ATV, IDV, and NFV*". In this example, the parser returns a parse tree in the form of noun phrase (NP) instead of a clause form. However, in some cases, the system can still extract relations by applying rule 2 on noun phrase such as this example.

Nonspecific rules: The second major source of errors is due to cases where the rules are not covered (18/62), as for instance in: "*Additional insertion of M184V into the zidovudine background doubled the resistance, whereas 44/118 did not lead to a further increase*". In such cases, the distance from relation word to keyword is longer than the defined values set in the rules. This can be corrected by relaxing the defined rules; however, this would also mean reducing the precision.

Semantic problems: In some cases, the errors were caused by semantic problems (12/62). This occurred when a relation is implied or hyponyms are used. For example: "*The PI mutation I50L causes clinically relevant resistance and increased susceptibility to atazanavir and other PIs respectively*".

There were only a very few cases where the source of error was caused by negation or anaphora resolution. Currently we do not take those sparse situations into account.

3.3.3 Relation combination performance

We extracted relations from all candidate sentences of the collected abstracts and obtained 2,434 extracted relations. After grouping relations and dropping relations which belong to response and association groups, we obtained 612 <mutation, drug> pairs. However, among these, there were only 163 pairs containing more than or equal 3 extracted relations. We selected 63 <mutation, drug> pairs for training the logistic regression function. The remaining 100 pairs are used to predict resistance values. To evaluate the results of the relation combination process, we selected relations of the 10 most common mutations. For each <mutation, drug> pair to be chosen as output relations, it is required that this pair has at least 3 extracted relations from the text. In addition, we have also calculated the resistance type based on three levels of resistance: susceptible (S), intermediate resistant (I) and resistant

(R) using the same method as proposed for two levels. Table 3.7 shows examples of the output of our system on K65R mutation and its related drugs.

The result in Table 3.7 shows that the output of our system has the same resistance type compared with the Stanford HIVDB on K65R mutation. There are 3 relations that have a different resistance type between the Stanford HIVDB and the RegaDB. This discordance is due to the fact that there are cases where RegaDB does not take into account the single <mutation, drug> pairs, therefore the RegaDB gives as an output “susceptible”, e.g. in case of <K65R, D4T> and <K65R, ABC> pairs. In contract, our system and the Stanford HIVDB do have evidence for these resistance pairs.

Table 3.8 shows a summary of the output results of the 10 common mutations, which account for 33% of extracted relations (615 instances over 54 <mutation, drug> pairs) and cover 3 common drug classes (PI, NRTI, NNRTI). The results are compared manually with the Stanford HIVDB system. The percentage of the agreement between two systems based on two levels of resistance (S, R) are 85%, and based on three levels of resistance (S, I, R) are 76%. By following the reference links provided by the Stanford HIVDB, we discovered that the main reason for the differences of the output between our system and the Stanford HIVDB is that there are many relations which can only be found from full texts, not from abstracts. In addition, the Stanford HIVDB also uses experimental data (e.g., n-fold value of resistance), while our system only uses pure text to synthesise the relations.

Table 3.7. Prediction results of mutation K65R and its related drugs

Mutation	Drug	Resistance type	HIVDB	REGADB
K65R	3TC	I	I	I
K65R	ABC	I	I	S
K65R	AZT	S	S	S
K65R	D4T	I	I	S
K65R	DDI	I	I	I
K65R	FTC	I	I	I
K65R	TDF	I	I	R

The results of K65R mutation and its drug resistance value calculated by the system compared with the result of the Stanford HIVDB based on three levels of resistance: susceptible (S), intermediate resistant (I), and resistant (R). In addition, we also provided the output of the RegaDB to show the differences between the expert systems

Table 3.8. Summary of the prediction results of the 10 most frequent mutations and their related drugs extracted from text compares with the HIVDB on two levels and three levels of resistance: susceptible (S), intermediate resistant (I), and resistant (R).

Mutation	Drugs	Agreement with the Stanford HIVDB output (%)	
		Two levels: S, R	Three levels: S, I, R
I84V	ATV, IDV, LPV, NFV, SQV,TPV	6/6	6/6
K103N	AZT, DLV, EFV, NVP	3/4	6/6
K65R	3TC, ABC, AZT, D4T, DDI, FTC, TDF	7/7	7/7
L74V	3TC, ABC, AZT, D4T, DDI	3/5	3/5
L90M	ATV, IDV, LPV, NFV, SQV	5/5	4/5
M184V	3TC, ABC, AZT, D4T, DDI, EFV, FTC, NVP, TDF	6/9	7/9
M46I	ATV, IDV, NFV, SQV	3/4	2/4
Q151M	3TC, ABC, AZT, D4T, DDI	5/5	3/5
V82A	IDV, LPV, NFV, SQV	4/4	3/4
Y181C	AZT, D4T, DLV, EFV, NVP	4/5	3/5
Over all		**85%**	**76%**

Furthermore, there are many pairs of <mutation, drug> where the number of extracted relations is below the threshold we have set, so these pairs are not considered by the system and do not appear in the output results. However, we also discovered that our system can extract new relations that do not appear in the Stanford HIVDB as shown in example of K65R mutation above.

Atomic value vs. group values: The atomic relations obtained by splitting a group of mutations from original relations are also the cause for disagreement between the output of the system and the Stanford HIVDB result. This was due to the fact that, in some contexts, the resistance only occurs if these mutations come together, but does not occur in a single mutation. For future work, we will take this issue into account.

PubMed abstracts are certainly a good source for extracting causal relations on HIV drug resistance; however, the number of extracted relations from abstracts is relatively low, only 5% of abstracts may potentially provide evidence for drug resistance. Therefore a more advanced form of publishing as proposed in [20] might provide a better solution for collecting data. In addition processing of full texts can be considered. The system can be used as annotator to extract relations from full text articles, the results are then considered as raw relations, which can be evaluated by experts. This system can save experts a significant amount of time otherwise spent finding relevant sentences which provide evidence for drug resistance. For convenience, the system provides summarized data and original texts, from which the relations were extracted, to support the experts in the verification of the results.

3.4 Conclusions

We have proposed a new method for extracting causal relations between drugs and mutations. By using grammatical relations and grouping mutations and drugs, we can reduce the syntactic variants of relations into two main forms, thus making the process of extracting relations much easier and less error prone.

We have also described a method to combine extracted relations which combines the manner of each individual relation and deals with contradictory relations in order to determine resistance type for each <drug, mutation> pair. The output of the relation combination shows promising results with 85% and 76% agreement to the Stanford HIVDB on two and three levels of resistance respectively. Furthermore, the system can also provide new relations and additional sources of evidence to analyze the discordance between expert systems.

The proposed algorithm uses publicly available NLP tools. Therefore, it is easy to setup a similar system, and it is suitable for extracting relations in case where an annotated corpus is not available. The algorithm can also be applied to extract other types of relations in which entities have a distinct category such as gene-protein, gene-disease, and disease-mutation. In such cases, the system needs to provide a list of relation words, manner words, and Name Entity Recognition (NER) module.

The performance of the system is good in terms of run-time. The system processed 129,448 abstracts on a Centrino Duo 1.8 GHz laptop in 35 minutes, of which 97% of the time was used by the parser, 2% was used for filtering and simplification of the sentences, and 1% of the time was used for the actual extraction and combination of the relations. The system is clearly capable to be used in large-scale relation extraction experiments. The system is used in the Virolab project (www.virolab.org) to preselect the most relevant novel resistance data from literature and present those to virologists and medical doctors for further evaluation.

Chapter 4. A hybrid approach to extract protein-protein interactions

Abstract

Protein-protein interactions (PPIs) play an important role in understanding biological processes. Although recent research in text mining has achieved a significant progress in automatic PPI extraction from literature, performance of existing systems still needs to be improved.

In this chapter, we present a novel algorithm for extracting PPIs from literature which consists of two phases. First, we automatically categorize the data into subsets based on its syntactic properties and extract candidate PPI pairs from these subsets. Second, we apply support vector machines (SVM) to classify candidate PPI pairs using features specific for each subset. We obtain promising results on five benchmark datasets: AIMed, BioInfer, HPRD50, IEPA, and LLL with F-scores ranging from 60% to 84%, which are comparable to the state-of-the-art PPI extraction systems. Furthermore, our system achieves the best performance on cross-corpora evaluation and comparative performance in terms of speed.

[*] This chapter is based on Bui et al., “A hybrid approach to extract protein-protein interactions”. *Bioinformatics*. **27**, 259-265 (2011).

4.1 Introduction

Extraction of protein-protein interactions (PPIs) from literature is an important research topic in the field of biomedical natural language processing (NLP) [50] Numerous PPIs have been manually curated and stored into databases, such as BIND, HDPR, and IntAct. However, this task has been proven time and resource-consuming. As a consequence, most data on PPIs can only be found in literature [24].

Several approaches for extracting PPIs from biomedical text have been reported. These methods range from co-occurrence to more sophisticated machine learning (ML) systems augmented by NLP techniques. Co-occurrence is the simplest approach, which results in high recall but low precision. Rule or pattern based approaches can increase precision but significantly lower recall. In addition, these rule sets are derived from training data and are therefore not always applicable to other data they are not developed for [25, 34].

Recently, many machine learning based methods have employed NLP techniques such as shallow parsing or full parsing [43, 87, 118, 119]. Since full parsing produces more elaborate syntactic information than shallow parsing, PPI extraction systems based on full parsing can potentially yield better results [26]. The output of the parser can be represented either as constituent trees or dependency trees. In this case, the PPI extraction task is treated as a binary classification problem which requires a formal protein pair representation and a suitable machine learning method. A protein pair (an instance) can be represented by a set of features, which are derived from the sentence or its syntactic structure. A machine learning method is then used to distinguish between positive and negative instances [45, 120].

Many linguistic features and machine learning methods have been proposed for the PPI extraction task. Based on feature types, these approaches can be divided into three groups. The first group focuses on lexical and word context features. Bunescu and Mooney [53] designed a subsequence kernel which uses the following information in a sentence: before the first protein, between two proteins, and after the second protein, and combined these features to obtain patterns. Giuliano et al. [87] extended this approach by using a bag-of-words and adding a local context kernel. The second group exploits syntactic features of a sentence. Sætre et al. [90] used various syntactic path features and context features related to words before, between, and after two interacting entities. Katrenko and Adriaans [88] proposed a method based on information found in the pre-defined levels of the dependency trees, such as local dependency contexts of the protein names and tree’s roots. Kim et al. [51] enhanced previous work by proposing a walk kernel which explores the shortest dependency path between two proteins and a modified dependency tree with the parts-of-speech (POS) features. As an alternative to previous approaches, Airola et al. [58] introduced an all-paths graph kernel. They represented a sentence with a dependency graph and considered dependencies connecting two entities outside the shortest path as well as on the shortest path.

Along with the proposed methods, Fayruzov et al. [100], Niu et al. [95], and Van Landeghem et al. [96] also studied individual impact of a variety of feature types on the PPI extraction task. In addition, a study of Miyao et al. [26] has shown that the accuracy of syntactic parsers also contributes to the overall performance of the PPI systems. To compensate for the limitations of each individual feature set and parser errors, Miwa et al. [50, 63] proposed a method that combines all the lexical and parsing features using multiple kernels and parsers. Their system achieved the state-of-the-art performance on a number of benchmark data sets. However, the studies by Fayruzov et al. [100] and Kim et al. [98] also demonstrated that if two feature types have overlapping rather than complimentary effects, dropping one of them can result in a computationally more efficient method and potentially make a mining algorithm more robust. This argument was also confirmed in the work of Miwa et al. [63] who showed that excluding the bag-of-words feature leads to better performance on the AIMed corpus.

Although many approaches have been proposed in the past, the problem of finding the most suitable features for extracting PPIs remains. Adding more features might sometimes improve performance, but they can introduce noise in other cases, or require more computational resources [98]. In this chapter, we propose a novel method that consists of two phases. First, we apply semantic rules to partition the data set into subsets according to its semantic properties and extract candidate PPI pairs from these subsets. Second, we introduce enhanced feature sets for use with a machine learning classifier to classify these extracted PPI pairs.

To the best of our knowledge, this is the first method that categorizes data into subsets and selects the most appropriate features for each subset. As a result, we increase the robustness of the learning method and make it computationally effective. Our study only focuses on the PPI extraction task with an assumption that relevant named entities were manually annotated.

4.2 System and methods

The workflow of the proposed system is as follows:

1. Text preprocessing
2. Extracting candidate PPI pairs
3. Classifying extracted PPI pairs

4.2.1 Text preprocessing

Text preprocessing includes converting protein names using a predefined rule set, filtering out input sentences with only one protein, splitting input sentences contain multiple clauses, and parsing sentences using the Stanford lexical parser[1].

Processing protein names

In order to improve accuracy of the parser, we replace all mentioned protein names with a place holder, i.e. PRO1, PRO2 (we refer to them as PRO*). We define a rule set to resolve the problem with embedded protein names (e.g., AIMed corpus), protein names that share prefix or suffix (e.g., AIMed and BioInfer corpora), and protein names including multiple positions. After this process, the number of proteins in the sentence is not changed. The list of original protein names for each sentence is maintained for reference purposes.

Replacing parenthetical remarks and splitting a sentence

If no word inside parentheses is a protein name, then all words and the parentheses are removed. In case the sentence consists of multiple clauses, the system splits it into clauses. The resulting sentence is referred to as a *simplified sentence*.

In the last step of text preprocessing, a simplified sentence that contains at least two protein names is analyzed with the Stanford lexical parser to produce a syntactic tree. All parse trees are stored in a local database for later use. An example of a sentence and its output after text preprocessing step is given below:

AIMed.d101.s852: The bZIP domains of Fos and Jun mediate a physical association with the TATA box-binding protein.

Its simplified sentence: The bZIP domains of PRO0 and PRO1 mediate a physical association with the PRO2.

4.2.2 Extracting candidate PPI pairs

Previous studies [48, 55] and our study in chapter 3 have shown that, in biomedical text, relation between two entities (protein-protein, protein-gene, drug-mutation, and others) can be expressed in the following abstract forms:

Form 1: PRO_i *word** REL (verb) *word** PRO_j

Example: *PRO1 interacts with PRO2*.

Form 2: PRO_i *word** REL (noun) *word** PRO_j

Example: PRO1 *has a weak interaction with PRO2*.

Form 3: REL (noun) *word** PRO_i *word** PRO_j

1 http://nlp.stanford.edu/software/lex-parser.shtml

Example: *interaction between PRO1 and PRO2*.

Here, REL is a cue word (interaction, inhibit, etc.) and can be a noun or a verb. *word** are tokens between PRO* and REL. PRO_i and PRO_j can be any protein pair and j>=i+1 (e.g., <PRO1, PRO2>). In addition, a closer look at the annotated corpora (e.g., AIMed and BioInfer) reveals that we can define two more forms:

Form 4 (*compound form*): PRO_i/PRO_{i+1} or PRO_i PRO_{i+1} or PRO_i-PRO_{i+1}, if two entities appear in the sentence with the patterns above, they seem to have interaction.

Example: *PRO1/PRO2 binding; PRO1-PRO2 compound*.

Form 5 (*complex form*): PRO_i *word** PRO_j *word** REL

Example: *PRO1, PRO2 interact*; *in PRO1, PRO2 complex*.

Form 4 is expressed as a pattern and requires an exact match. For other forms, there might be one or more tokens between <PRO_i, REL>, <REL, PRO_j> or <PRO_i, PRO_j>. Based on these basic forms, we now map the semantic relations of these forms into parse trees. For convenience, we define the following patterns:

a) *Full clause*: S < ((NP << PRO*) $++ (VP << PRO*))

b) *Partial clause*: VP < ((NP << PRO*) $++ (S < (VP << PRO*)))

c) *Sub-clause*: S < ((NP << PRO*) $++ (VP < (S|SBAR << PRO*)))

d) *NP pattern*: NP < ((NP <<# REL) $++ (PP << PRO*))

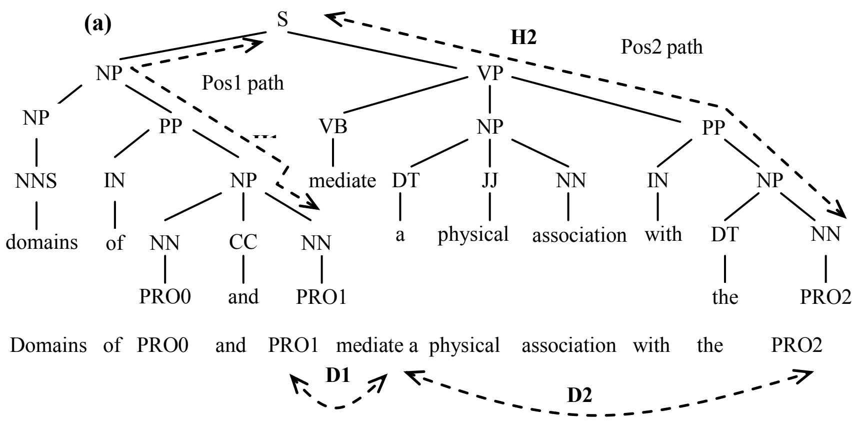
Here S (clause), NP (noun phrase), VP (verb phrase), PP (prepositional phrase), and SBAR (sub-clause) are constituents of the parse tree, PRO* is a place holder for a protein name, and REL is a cue word of the input sentence. All patterns above are written using the Tregex syntax[2], which is developed within the Stanford parser package. Figure 4.1 illustrates some parse trees with the patterns mentioned above, e.g., the parse tree in Figure 4.1a shows a *full clause*, Figure 4.1b shows a *NP pattern* and Figure 4.1c shows a *sub-clause*.

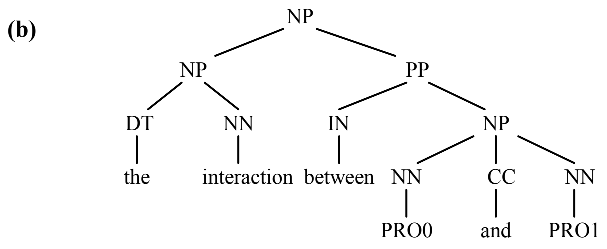
Relation list

We created a relation list by combining the relation lists used in the previous work by Chowdhara *et al.* [111], Fundel *et al.* [55], and Niu *et al.*[95].

In the following section, we describe the procedures and algorithm to extract candidate PPI pairs from a parse tree or a simplified sentence:

[2] http://nlp.stanford.edu/software/tregex.shtml





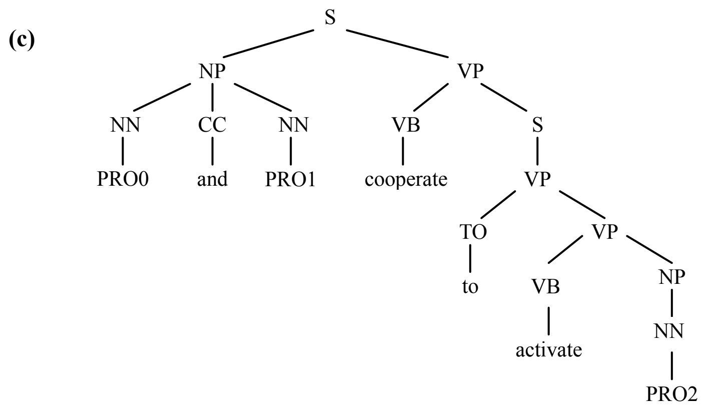


Figure 4.1. Examples of parse trees output from the Stanford lexical parser. Figure 4.1(a) shows some features (D1, D2, H1, H2) and paths from the join node connecting a given protein pair (PRO1, PRO2).

Procedure for Form 1: This procedure requires a parse tree which contains *full clause*, *partial clause*, or *sub-clause* patterns. The procedure is as follows:

a. Check a head word of VP that corresponds to satisfied pattern. If this verb belongs to the relation list, use it as REL.

b. Create a *right_keys* list: use all keys in VP as *right_keys*.

c. Create a *left_keys* list: use all keys in NP if a satisfied pattern is *full clause* or *sub-clause*. If a pattern is *partial clause,* then find a sister NP immediately preceding VP.

d. Form candidate PPI pairs: enumerate the *left_keys* and *right_keys* to compose a triple <PRO1-REL-PRO2>.

Procedure for Form 2: Form candidate PPI pairs from a simplified sentence if they satisfy the following form: PRO* – REL – PRO*.

Procedure for Form 3: This procedure requires a parse tree which contains *NP pattern*. The procedure is as follows:

a. Check a head word of NP that corresponds to satisfied pattern. If this noun belongs to the relation list, use it as REL.

b. Find a proposition as a splitter in pattern's PP phrase in order to create *left_keys* and *right_keys* lists.

c. Form candidate PPI pairs: enumerate the *left_keys* and *right_keys* to compose a triple <PRO1-REL-PRO2>.

Procedure for Form 4: Form candidate PPI pairs from an input sentence if they satisfy any of the following forms: PRO*/PRO*; PRO* PRO*; PRO*-PRO*.

Procedure for Form 5: Form candidate PPI pairs from the simplified sentence if they satisfy the following form: PRO*- PRO* - REL*,* and if the distance from the first PRO* in the pattern to the REL is shorter than 5 tokens.

For these procedures, the extracted candidate PPI pairs obtained from the same procedure are grouped together i.e. output from Form1 are grouped into group 1. Among these groups, group 2 is a special case of group 1, with the purpose to recover PPI pairs that the group 1 failed to extract due to the parser errors [26]. As a consequence, in some cases, a PPI pair may belong to more than one group. Therefore, the order in which patterns are applied is important (as described in the Algorithm 4.1 below).

Algorithm 4.1. // Algorithm to extract candidate PPI pairs.

Input: simplified sentence, parse tree, and the relation list

```
Output: list of candidate PPI pairs of corresponding groups
Init: used_list = null // store extracted pairs to avoid overlap since one pair can satisfy more than one pattern.
For a list of sub-sentence/parse tree
    For form from 1 to 5
        Extract candidate pairs from parse tree or simplified sentence
        If candidate pairs found
            For each candidate pair
             Check whether this pair in the used_list
              If not found
                    Store this pair to the corresponding group
                    Store this pair to the used_list
                End if
            End for
        End if
    End for
End for
```

Features

In this section we propose feature sets for machine learning classification. Our feature sets are the modification and combination some of the features that were previously proposed by Chowdhary *et al.* [111], Giuliano *et al.* [87], Miwa *et al.* [45, 50], and Niu *et al.* [95]. For each candidate PPI pair extracted from an input sentence, the system outputs a triple e.g., <PRO_i, REL, PRO_j> then the following features are generated:

Keyword: is a relation word (REL) from the extracted triple. In addition, we also count the number of protein names (C1) and relation words (C2) in each simplified sentence.

Distance: We use two features: D1, D2 (see Figure 4.1a) to measure the distance (number of words/tokens) between PRO1-REL and REL-PRO2 (or between REL-PRO1 and PRO1-PRO2 for group 3).

Height: We use two features: H1, H2 (see Figure 4.1a) to measure the distance from the joint node connecting PRO1 and PRO2 in the parse tree. These features are similar to D1 and D2 except that they measure the number of nodes on the paths from a local root to PRO1 and PRO2.

POS: We use two lists: Pos1, Pos2 (see Figure 4.1a) to store POS and syntactic features from the joint node connecting PRO1 and PRO2 in the parse tree, respectively.

Lexical: This feature is a modification of a bag-of-words (BOW). Instead of using all tokens, we only consider tokens that belong to the *relation list* and a list of prepositions: *and, or, by, through, in, of, to,* and *between*. If a token is a protein (PRO*), then its value is replaced with "KEY". We use four lists of tokens:

L1: a list of tokens between PRO1 and REL, or between REL and PRO1 for group 3.

L2: a list of tokens between REL and PRO2, or between PRO1 and PRO2 for group 3.

L3: a list of tokens before PRO1.

L4: a list of tokens after PRO2.

Selection of features for each individual group

The important benefit of partitioning data into subsets is that we can select the most appropriate features for each subset. Let us consider the following two cases: a PPI pair in form 4 (compound form) and a PPI pair in form 1 (full clause). For a PPI pair in a compound form, e.g. PRO1-PRO2, the shortest path features proposed in the previous work become useless because no feature can be extracted from this path. In this case, the bag-of-word features seem to be the most appropriate ones. In contrast, for the PPI pairs in form 1, e.g. PRO1 interacts with PRO2, the shortest path, and the tokens between PRO1 and PRO2, play an important role. Based on the properties of each group of extracted PPI pairs, we manually select features that are potentially suitable for that group. Table 4.1 shows the list of features corresponding to each group. Furthermore, we also use the attribute selection function from the WEKA machine learning package [49] to determine the length of the POS and lexical lists, which are limited to maximum 6 attributes.

Machine learning method

Support vector machines (SVM) have been widely used in the PPI extraction task, and has shown competitive performance over other learning methods ([63, 98]). In this work, we use SVM classifier with a default RBF kernel and tune the complexity parameter C by using *CVParameterSelection* function from the WEKA. All individual features mentioned above are combined into a single feature vector for the classification task. Depending on each group of PPI candidates, the number of features ranges from 12 to 18. Table 4.1 summarizes features used for each group of the PPI candidates. For example, for a candidate PPI pair <PRO1, PRO2> extracted from group 1 (as shown in Figure 4.1a), the following features are generated:

REL: mediate; D1: 1, D2: 6; C1: 3, C2: 2; H1: 5, H2:4; L2: association, with, null; Pos2: VB, PP, NP, NN, null, null, true.

Table 4.1. List of features and their usage for each group

Feature	Keyword	Distance	Height	POS	Lexical
Group 1	REL, C1, C2	D1, D2	H1, H2	Pos1,2	L2, L2
Group 2	REL	D1, D2	H1, H2		L1, L2
Group 3	REL, C1, C2	D1, D2	H1, H2	Pos1,2	L2
Group 4	C1, C2				L3, L4
Group 5	REL C1, C2	D1, D2			L3, L4

4.3 Results and Discussion

4.3.1 Data sets

We use five corpora[3] which have been converted into a unified format and are provided by Pyysalo et al. [29]: AIMed, BioInfer, HPDR50, IEPA, and LLL. Table 4.2 shows the quantitative information of all 5 corpora.

Table 4.2. Statistics on 5 corpora

Corpus	AIMed	BioInfer	HPRD50	IEPA	LLL
Positive pairs	1000	2534	163	335	164
All pairs	5834	9666	433	817	330

4.3.2 Evaluation methods

We perform two types of evaluation, a single corpus test and a cross-corpora test. In the single corpus test, we evaluate the performance of the proposed method by 10-fold abstract-wise cross-validation (CV), and use the one-answer-per-occurrence criterion [63]. We split the corpora as recommended by Airola *et al.* [58] in order to directly compare our results against performance of other systems. In the cross-corpora test, we conduct two experiments. First, we use one corpus for training and

[3] http://mars.cs.utu.fi/PPICorpora/GraphKernel.html

test the system on the four remaining corpora. Second, we use four corpora (ALL) for training and test on the remaining corpus.

4.3.3 Performance of PPI extraction algorithm

Table 4.3 shows the results of the PPI extraction algorithm on 5 corpora. In order to calculate recall for each corpus, we use the number of original positive pairs from Table 4.1 (e.g., for AIMed, TP+FN=1000, for BioInfer, TP+FN=2534). It is worth noting that, in some systems [58, 98], self-interaction pairs are removed prior to the evaluation therefore the F-score can be higher if we adopt these settings.

Table 4.3 also presents the properties of each extracted group in each corpus and the difference between corpora. Clearly, group 1 is the most common among all corpora and accounts for more than 50% of true positive pairs of all corpora except for BioInfer. The first three groups are also common for all 5 corpora and account for more than 80% of the extracted pairs. However, group 4 and 5 are more corpus-specific and can be found mostly in AIMed and BioInfer corpora. This information provides more insight for understanding the properties of 5 corpora studied previously by Pyysalo *et al.* [29].

Table 4.3. Results of PPI extraction algorithm on 5 corpora

Corpus	AIMed		BioInfer		HPRD50		IEPA		LLL	
	TP	FP	TP	FP	TP	FP	TP	FP	TP	FP
Group 1	500	701	849	648	101	38	170	78	106	13
Group 2	143	112	215	154	14	10	26	26	2	4
Group 3	113	322	330	529	17	23	92	61	32	11
Group 4	22	55	170	73	0	8	4	3	0	2
Group 5	39	192	134	182	6	5	3	6	0	1
Total	**817**	**1382**	**1698**	**1586**	**138**	**84**	**295**	**174**	**140**	**31**
Recall	81.7%		67%		84.7%		88.1%		85.4%	
Precision	37.2%		51.7%		62.2%		62.9%		81.9%	
F-score	**51.1%**		**58.4%**		**71.7%**		**73.4%**		**83.6%**	

The results in Table 4.3 demonstrate that the proposed algorithm is capable of extracting of more than 80% of positive pairs (recall >80%) on AIMed, HPRD50, IEPA, and LLL corpora. This means that these four corpora share some common patterns or the same annotation policy. However, our method can only extract 67% positive pairs from BioInfer. To find out why our algorithm has lower recall on the

BioInfer corpus, we examine sample sentences from BioInfer which the system failed to extract. By analyzing these sentences, we discovered that in some cases, BioInfer has a special annotation policy. Consider, for example, the following sentence:

BioInfer.d436.s0: *Moreover, ectopic expression of PRO0 and PRO1 can stimulate the PRO2 promoter in an PRO3-dependent manner, and this is inhibited by coexpression of the PRO4 (PRO5) PRO6.*

For this input sentence, our system extracts only 4 true positive pairs, but the number of positive PPIs pairs provided by the BioInfer corpus is 11 (see sentence *BioInfer.d436.s0* in BioInfer corpus for more detail). Therefore, to increase recall on BioInfer, additional study of this corpus is needed.

Table 4.3 also shows that the performance of algorithm 1 is comparable with other machine learning based systems. Moreover, it outperforms the best rule-based system on all 5 corpora (see [34]) for more details).

4.3.4 Single corpus evaluation

With the extracted PPI pairs obtained in the previous step, we use a standard SVM classifier to classify them. Comparing with the original data from Table 4.1, the number of positive instances and negative instances of the AIMed and BioInfer corpora is quite balance (see Table 4.3). To calculate the F-score, we accumulated all true positives (TP) and false positive (FP) pairs from the 10-fold CV test (reported by WEKA via the confusion matrix). For TP+FN values, similarly to section 3.3, we use the original positive pairs of each corpus in Table 4.1. For the IEPA, HPRD50, and LLL corpora, however, we only use the first three groups of extracted pairs because the number of candidate pairs in the group 4 and 5 is too small for the 10-fold cross-validation.

Table 4.4. Results of the 10-fold abstract-wise cross-validation on 5 corpora. ΔP and ΔF show the increase in precision and F-score and ΔR shows the decrease in recall compared to their corresponding values in Table 4.3. The values are shown in percentage (%).

Corpus	Precision	ΔP	Recall	ΔR	F-Score	ΔF
AIMed	55.3	18	68.5	(13.2)	61.2	10
BioInfer	61.7	10	57.5	(9.5)	60.0	2
HPRD50	70.2	8	77.9	(6.8)	73.8	2
IEPA	67.4	5	83.9	(4.2)	74.7	1
LLL	84.1	2	84.1	(1.3)	84.1	1

Table 4.4 shows the performance of the classifier on five corpora. The results indicate that by using our feature vectors, the classifier can further boost the performance of the overall system. When comparing precision on each corpus before and after applying classification, we can see a significant increase in precision (ΔP values). Among all corpora, the AIMed corpus gains the most increase in precision with ΔP of 14%. Even on a small corpus like LLL, with already very high precision, the final precision still gains 2%. For each corpus, recall decreases after applying classification but the final F-scores increase for all corpora and range from 1% to 10%. Furthermore, Table 4.4 shows that our system achieves the best performance on three corpora: IEPA, HPRD50, and LLL using 10-fold cross-validation when compared to other results reported so far.

Table 4.5. Performance comparison between full dataset, filtered dataset and partitioned dataset using 10-fold CV on five corpora. Full dataset and filtered dataset use the same feature sets. The performance is measured by the F-score (%).

Corpus	AIMed		BioInfer		HPRD50		IEPA		LLL	
	P	F	P	F	P	F	P	F	P	F
Full dataset	51.8	52.5	56.5	58.7	65.7	68.6	66.0	72	78.9	83.6
Filtered dataset	55.8	55.2	63.2	59.6	76	76	67.8	72.9	85.2	84.4
Split dataset	55.3	61.2	61.7	60	70.2	73.8	67.4	74.7	84.1	84.1

To study the benefit of filtering data (obtained after applying rules on the full dataset) and partitioning data, we conduct the experiment in which the full dataset and the filtered dataset use all features in Table 4.1. However, in this experiment the REL feature is not available, D1 and D1 features are replaced by D1+D2, and L1 and L2 features are merged. Table 4.5 summarizes the performance of the system on the full dataset (all PPI pairs), filtered dataset, and partitioned dataset. The results from Table 4.5 show that when evaluating on filtering dataset, the system has better performance on all five corpora compared with full dataset. In addition, the performance further improve on three largest corpora (AIMed, BioInfer, IEPA) when we partition data and select feature specific for each sub-dataset. This clearly shows the benefit of our hybrid approach that combines rule with partition data.

Table 4.6 illustrates a comparison of our system (BKS) against recent work on the AIMed corpus, which is considered the *de-facto* benchmark for the PPI extraction task. However, the comparison may not be straightforward because these PPI systems can use different text preprocessing and learning methods. The results show that our system with its F-score of 61.2% is comparable to the state-of-the-art systems proposed by Miwa *et al*. [63] and Sætre *et al.* [45]. In addition, our approach significantly outperforms other methods based on one parser's output.

Table 4.6. Performance comparison with other systems on AIMed

System	Description	F-score (%)
Sætre [45]	Feature-based, two parsers	64.2
Miwa [50]	Multiple kernels, two parsers	60.8
Kim [98]	Walk-weighted subsequence kernels, one parser	56.6
Airola [58]	All-paths graph kernel, one parser	56.4
Niu [95]	Linear kernel, one parser	53.5
BKS	RBF kernel, one parser	61.2

4.3.5 Cross-corpora evaluation

In addition to the standard 10-fold cross-evaluation, recent studies also suggested to test existing approaches to PPI extraction using cross-corpora. This type of evaluation can show whether a system trained on one corpus can perform equally well on other corpora, with an assumption that these corpora addressed the same PPI task. In this setting, one corpus is used for training and the remaining corpora are used for testing. For the evaluation, we only use the two largest corpora, i.e. AIMed and BioInfer, for training since the other corpora are too small after partitioning onto 5 groups. Furthermore, we conduct the experiment proposed by Fayruzov *et al.* [100] and Kabiljo *et al.* [34], in which four corpora are used for training and the remaining corpus for testing.

Table 4.7 shows the results of the cross-corpora evaluation on five corpora. Here, the columns correspond to the training set and the rows correspond to test sets. Table 4.7 illustrates that our system achieves the best performance when BioInfer is used as the training set. Note that we do not use data from group 4 and group 5 due to theirs corpus-specific properties, therefore the performance might be higher if these groups are used. This finding is also consistent with the evaluation results by Miwa *et al.* [50] and Airola *et al.* [58]. In their work, better performance is also obtained with BioInfer being used for training.

Interestingly, Table 4.7 shows that even though the F-score on each corpus in the cross-corpora test is lower than F-score in the single corpus test, the precision is significantly higher when compared against the initial values in Table 4.3. This means that performance on all corpora can be boosted when precision is given a priority. This also implies that the classifier is able to learn from cross-corpora training data. In other words, the proposed feature vectors are compatible across the corpora.

Table 4.7. Results of the cross-corpora test on five corpora. Columns correspond to training and rows correspond to testing corpora. ALL refers to a situation where four corpora are used for training and the remaining corpus for testing. Precision (P) and F-score (F) are shown in percentage (%).

Corpus	AIMed		BioInfer		ALL	
	P	F	P	F	P	F
AIMed	-	-	44.4	55.2	44.1	55.1
BioInfer	57.1	54.4	-	-	64.3	50.5
HPRD50	67.0	72.3	69.9	74.0	68.8	70.8
IEPA	67.3	73.6	68.6	73.8	70.1	72.9
LLL	85.7	83.0	85.0	82.4	86.0	80.1

Table 4.8 shows the performance of our system (BKS) in the cross-corpora setting compared with other approaches. The results in Table 4.8 demonstrate that BKS outperforms others when AIMed and BioInfer corpora are used as training sets. In addition, for all evaluation tests, we use the same settings for training corpora except for the complexity term (C parameter of the RBF kernel). This is one step closer to the real world situation.

Table 4.8. Performance comparison with other systems using cross-corpora evaluation. Columns correspond to training and rows correspond to test sets. The performance is measured by the F-score (%).

	AIMed				BioInfer			
	BKS	kBSP [33]	Miwa [50]	Airola [58]	BKS	kBSP [33]	Miwa [50]	Airola [58]
AIMed	-	-	-	-	**55.2**	40.3	49.6	47.2
BioInfer	**54.4**	24.8	53.1	47.1	-	-	-	-
HPRD50	**72.3**	51.0	68.3	69.0	**74.0**	69.8	68.3	63.9
IEPA	**73.6**	36.7	68.1	67.4	**73.8**	72.4	71.4	68.0
LLL	**83.0**	44.4	73.5	74.5	**82.4**	80.6	76.9	78.0

4.3.6 Performance time

Another important aspect in PPI extraction is the computational time taken to train and test the system. A previous study by Van Landeghem *et al.* [91] has shown that in order to reduce the number of computational resources used by machine learning, one has to consider a trade-off between performance (in this case measured by the F-score) and the performance time required by ML. Despite this fact, only few systems report on how many computational resources their systems use for training and testing. Miwa *et al*. [63] demonstrate that by using suitable features as well as a learning method, they can improve the performance of their previously proposed system [50]. In addition, Fayruzov *et al.* [100] have pointed out that the more kernels the system uses, the more computational resources are needed.

Table 4.9 shows the running time of our system on the full dataset compared against partitioned dataset of the AIMed corpus. On the partitioned dataset, the system runs faster on both implementations of RBF kernel. In addition, since our method partitions data into 5 groups, we also test it in a node with 8 CPUs (Xeon 3.0 GHz). In this test, we use the RBF kernel from LibSVM and the classifier is run in parallel. The maximum time used by the system is 12 seconds. This means that our system not only runs fast on a single PC, but can also be used in parallel, which is particularly suited for large-scale experiments.

Table 4.9. Performance time for 10-fold CV of full and partitioned dataset of AIMed corpus. These values exclude time for parsing and text preprocessing. Two implementations of RBF kernel from WEKA and LibSVM[4] are used. The experiment is run on a PC with an Intel 3.2GHz processor and 4GB of RAM

Dataset	WEKA - RBF	LibSVM - RBF
Full dataset	10812 seconds	194 seconds
Partitioned dataset	77 seconds	39 seconds

4.4 Conclusions

In this paper, we propose a novel method for extracting PPI pairs from literature. Our approach combines the strength of both syntactic rules and machine learning classification. The evaluation on five benchmark corpora has shown that our system achieves results comparable with the best PPI extraction methods on a single corpus. It outperforms other systems on cross-corpora test and has fast running time.

[4] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

The proposed method consists of two phases: partitioning data into subsets then extract candidate PPI pairs from these subsets, and classifying extracted PPI pairs. The advantages of this method are four-fold. First, it filters out a significant amount of negative (non-interaction) PPI pairs, thus balancing the number of positive and negative pairs in training data. Second, by partitioning data into subsets, we can select the most appropriate features for each subset, which potentially improves the final performance. Third, our system only uses a small set of features and therefore performs the best in terms of performance time. Finally, the classifier can be run in parallel on each subset, which is desirable for the large- scale experiments.

In addition, our method uses five syntactic rules therefore it is generic and can be easily applied to new data sets. Furthermore, it is easy to set-up because it only uses publicly available NLP tools and a standard machine learning package. In addition, the proposed method can also be extended to extract new relation types in biomedical text, e.g. complex event extraction [118, 121].

Chapter 5. A robust approach to extract biomedical events from literature

Abstract

The abundance of biomedical literature has attracted significant interest in novel methods to automatically extract biomedical relations from texts. Until recently, most research was focused on extracting binary relations such as protein-protein interactions and drug-disease relations. However, these binary relations cannot fully represent the original biomedical data. Therefore, there is a need for methods that can extract fine-grained and complex relations such as biomedical events.

In this chapter we propose a novel rule-based method to extract biomedical events from text. Our method consists of two phases. In the first phase, training data are mapped into structured representations. Based on that, templates are used to extract rules automatically. In the second phase, extraction methods are developed to process the obtained rules. When evaluated on the Genia event extraction test set (Task 1) using '*Approximate Span/Approximate Recursive matching*' criteria, we obtain results with recall, precision and F-score of 42.63, 68.77 and 52.63%, respectively, which is comparable to the state-of-the-art systems. Furthermore, our system achieves superior performance in terms of speed.

[*] This chapter based on Bui & Sloot., "A robust approach to extract biomedical events from literature". *Bioinformatics* 2012. DOI: 10.1093/bioinformatics/bts487

5.1 Introduction

Automatic relation extraction is an important and established way of extracting information from biomedical literature [1, 107]. Many relation extraction approaches have been proposed to extract binary relations between biomedical entities from text such as protein-protein interactions [43, 45], drug-drug interactions [123], and causal relations on drug resistance [124]. However, such binary relations cannot fully represent the biological meaning of the original relations. As a consequence, there is a need to have a better representation that can exemplify complex relations extracted from text. Recently, this shortcoming is addressed in the BioNLP'09 Shared Task (BioNLP, hereafter) [121] by introducing the biomedical event extraction task, which aims to extract complex and nested events from text. An event is defined as follows: each event consists of a trigger, a type, and one or more arguments. Depending on the event type (e.g. simple, binding, regulation), an argument can either be a protein or another event (see [121] for more details). Figure 5.1 illustrates the complex and nested structures of the biomedical events. Figure 5.1a shows a *Negative_Regulation* event which has two events as the arguments, whereas Figure 5.1b shows a *Positive_Regulation* event which has one protein and one event as the arguments.

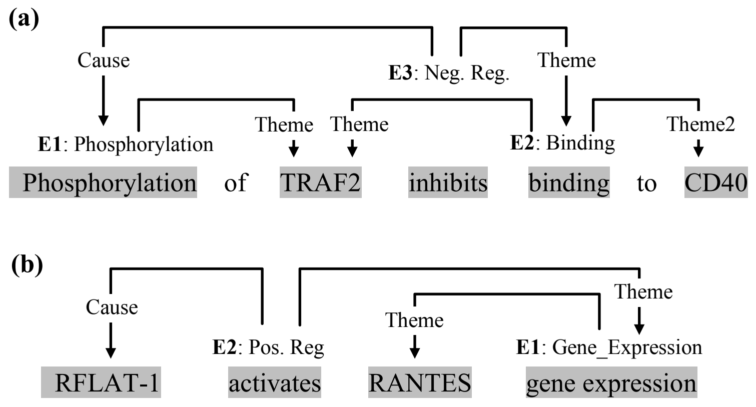


Figure 5.1. (**a**) Examples of three event types: simple (E1), binding (E2), and regulatory (E3) events. (**a**, **b**) Variants of regulatory event arguments: the event E3 in Fig. 1a has two events as the arguments whereas the event E2 in Fig. 1b has one protein and one event as the arguments

Several event extraction approaches have been proposed to the BioNLP'09 and BioNLP'11 challenges [60, 121]. In general, to extract events from a given sentence, two steps need to be carried out: identifying event triggers and determining their arguments. First, the sentence is typically preprocessed and converted into structured representations such as dependency parse tree and candidate event triggers are detected using a dictionary. Next, the candidate event triggers and the parse tree are used as the input for the event extraction process. The proposed approaches used to extract events can be divided into two main groups, namely rule-based and machine learning (ML)-based approaches.

Rule-based event extraction systems consist of a set of rules that is manually defined or generated from training data. To extract events from text, first each candidate trigger is validated to determine its event type. Second, the defined rules are often applied to the parse tree which contains the trigger to find relation between the trigger and its arguments [125–127]. Evaluation results from the BioNLP’09 and BioNLP’11 show that rule-based systems achieve high precision but low recall. In particular, the system proposed by Kilicoglu and Bergler [127] ranks second on the BioNLP’09 event extraction track (task 1).

ML-based systems deal with the event extraction task as a classification problem. In which, candidate event triggers are classified as true event triggers or not and which arguments attached to them are correct. Extraction methods proposed in the BioNLP’09 exploit various learning algorithms and feature sets to leverage the system performance [128, 129]. Methods proposed after the BioNLP’09 can be divided into two groups based on how event triggers and arguments are determined. In the first group, some systems adopt the workflow and feature sets proposed by Björne et al. [129] and later improved by Miwa et al. [130], in which the evaluation of the candidate triggers and the determination of their arguments are carried out independently. In this type of approach, errors made in the trigger detection step propagate into subsequent steps. Examples of such systems are [102, 103, 131]. To overcome this issue, the second ML-based group uses joint learning models in which event triggers and their arguments are jointly predicted. Systems belong to this group are [132–134]. Overall, ML-based systems achieve good results on both BioNLP’09 and BioNLP’11.

Apart from these two main approaches, there are a few systems that employ different methods. Móra et al. [135] propose a hybrid method that combines both rule- and ML-based approaches, Liu et al. [73] employ a system using sub-graph matching, Neves et al. [136] use case-based reasoning system, McClosky et al. [137] introduce a system using dependency graphs by extending the function of an existing dependency parser. Cohen et al. [138] propose a system using manually defined patterns while Nguyen et al. [75] employ a method that generates patterns automatically from training data. However, the performance of these systems is not comparable to the state-of-the-art ML-based systems.

In this chapter we propose a novel rule-based method to extract biomedical events from text. Our method differs in many ways from the existing methods. First, we define a new structured representation to express the relations between event triggers and their arguments. Next, we transform the input text into this structured representation using a shallow parser. We then map the annotated events from the training data into these structured representations, from that rules are extracted automatically by applying predefined templates. The obtained rules are unified to make them more flexible. Finally, we develop event extraction algorithms to process these unified rules. By employing the structured representation to decompose the nested and complex structures of events into simple syntactic layers, we can use simple methods to learn and process extraction rules. When evaluated on the test set, our system achieves a comparative performance in terms of precision and computational times. To the best of our knowledge, this is the first system of its type used to extract biomedical events from text.

5.2 System and methods

Our core method to event extraction consists of two phases: a learning phase and an extraction phase. In the learning phase, the system learns rules to extract events from training data. In the extraction phase, the system applies rules learned in the previous phase to extract events from unseen text. A text preprocessing step is used in both phases to convert unstructured text into a structured representation. In the following sections, we present these steps in more detail.

5.2.1 Structured representation

We define the following syntactic layers which form a structured representation to express the structures of biomedical events:

Chunk: is the base syntactic layer that can express an event (see Figure 5.2). A chunk consists of one or more tokens, taken from the output of a shallow parser (see section 2.5 for more detail). Four chunk types that are used to express events are: adjective, noun, verb, and preposition.

Phrase: consists of a list of chunks, connecting by preposition chunks. A phrase can express one or more events in which the event triggers and arguments should belong to different chunks. To reduce the variants when learning rules and to simplify the event extraction process, all noun chunks that are in the coordinative form are merged into one noun chunk. For example, the following chunks: [NP PRO1], [NP PRO2], [O and] [NP PRO3] are merged into one noun chunk [NP PRO1, PRO2, and PRO3].

Clause: consists of a verb chunk connecting with a left child and a right child. A child is a phrase but it can also be a chunk. A clause can express one or more events in which the event trigger belongs to the verb chunk and its arguments belong to the clause's children.

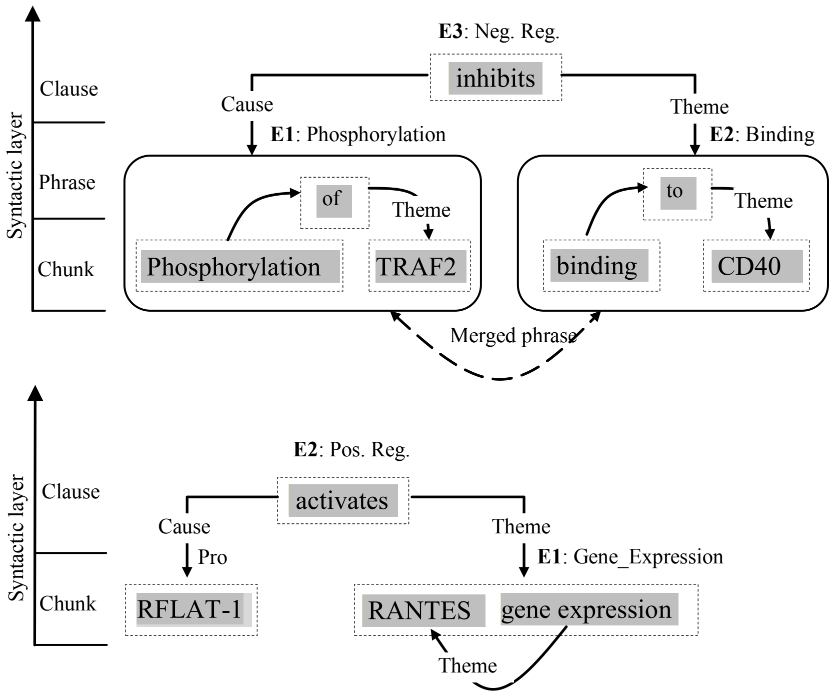


Figure 5.2. Structured representations for the events in Figure 5.1

Merged phrase: is a special noun phrase obtained by merging the verb chunk of a clause with its children. The merged phrase is used to express an event when its trigger and arguments belong to the left child and the right child of a clause. For example, the binding event E2 in Figure 5.1 needs a merged phrase to be expressed.

Relation between layers: An upper layer can query its direct lower layer to find arguments. The returned values of the query can be a list of proteins, events, or an empty list. For example, a phrase can query its right chunks. A clause can query its left and right children.

Representing biomedical events: with this structured representation, we can express most of biomedical events from training data. Of which, both event triggers and their arguments must belong to the same sentence. Figure 5.2 presents the events in Figure 5.1 using structured representations.

5.2.2 Learning rules

To learn extraction rules from the structured representations, we define a list of syntactic and semantic features to capture the underlying rules which govern the relations between event triggers and their arguments when expressed by structured representations. These features are divided into three groups which correspond to three syntactic layers: chunk, phrase, and clause. The descriptions of these features are as follows.

Features for events expressed in a chunk layer:

Frequency: counts frequency of an event trigger when expressed in chunk form.

Part-of-speech (POS): indicates which syntactic form (e.g. NN, ADJ) of that event trigger used in the chunk form.

Features for events expressed in the phrase layer:

Frequency: counts frequency of an event trigger when expressed in phrase form.

POS: indicates which syntactic form (e.g. NN, NNS, and VBG) of that event trigger used in the phrase form.

Preposition: indicates which preposition is used to connect the chunk which contains the event trigger with the chunk containing its first argument (*theme*), e.g. [NP *expression*] ***of*** [NP PRO1].

Distance: measures the distance (by chunk) from the chunk which contains the event trigger to the chunk containing its *theme*.

Preposition2: is used for binding and regulatory events. It indicates which preposition is used to connect the chunk which contains the *theme* to the chunk containing the second arguments (i.e. *theme2* for binding event or *cause* for regulatory event), e.g. [NP *binding*] **of** [NP *PRO1*] **to** [NP *PRO2*]; [NP *effect*] **of** [NP *PRO1*] **on** [NP *PRO2*].

Distance2: is used for binding and regulatory events. It measures the distance (by chunk) from the chunk which contains the event trigger to the chunk containing its second argument.

Cause order: is used for regulatory events. It indicates the relative position between the *theme* and the *cause* of an event. For example, consider the event: *effect* of *PRO1* on *PRO2* in which PRO1 is the *cause*. This feature helps determining whether the argument which is closer to the trigger is a *theme* or a *cause*.

Theme position**:** is used for binding events. It indicates the position of the *theme* is on the left or on the right of the event trigger. For example: PRO1 *binding* **to** PRO2, *interaction* **between** PRO1 **and** PRO2.

Theme 2 position**:** similar to theme position but for the second argument.

Features for events expressed in the clause layer:

Frequency: counts frequency of an event trigger when expressed in clause form

POS: indicates which syntactic form (e.g. VBZ, VBD) of the event triggers used in the clause form.

Passive: indicates whether the clause in the *active* or the *passive form*.

Theme position: indicates the *theme* is on the left child or on the right child when the clause in the *active form*.

Distance, Distance2: these features are similar to those of phrase form.

Beside these features, we use the following features to determine the number of arguments and argument types for binding and regulatory events.

Specific feature for binding events:

Theme2Count: counts frequency of an event trigger having *theme2*.

Specific features for regulatory events:

ThemePro: counts frequency of an event trigger having *theme* as a protein.
ThemeEvent: counts frequency of an event trigger having *theme* as an event.
CausePro: counts frequency of an event trigger having *cause* as a protein.
CauseEvent: counts frequency of an event trigger having *cause* as an event.

We create three templates, where each template consists of the features designated for a corresponding layer (i.e. chunk, phrase, and clause), to learn extraction rules by mapping each annotated event from the training data into a suitable layer. The learning procedure is shown in Algorithm 5.1.

Algorithm 5.1. Learning extraction rules from annotated events

Input: Sentence **S,** list of events **E** belongs to **S**, a dictionary **D**, map **H** for each event type, template **T**

Output: List of rules for each event type.

```
1  Convert S into structured representation R
2  For $e \in E$
3    If event trigger of $e$ in D
4      If $e$ can be mapped to layer $r \in R$
5        Fill $t \in T$ with features extracted from $r$
6        Generate key k for t
7        If $k \in H$
8          Update entry in H
9       Else
10 Store $k$ into H
```

The text conversion step used in this algorithm is described in section 5.2.5. During this learning process, some events are skipped if their event triggers do not belong to a dictionary which is prepared based on single-word event triggers collected from the training data. In addition, events are skipped if they cannot be mapped into the structured representation.

5.2.3 Rule combination

For each event trigger and its specific POS tag, we may obtain from one to three sets of extraction rules which correspond to three layers: chunk, phrase, and clause. The number of rule sets for each event trigger depends on its POS tags. For example, the *binding* trigger has three POS tags: ADJ, NN, and VBG therefore it may possess up to nine sets of extraction rules. Examples of a set of extraction rules of the *binding* trigger and its POS tag NN which learned from the phrase layer are shown in Table 5.1. The first row of this table is an extraction rule that learned from binding events such as the binding event E2 in Figure 5.2.

Table 5.1. Examples of a set of extraction rules of the *binding* trigger obtained from phrase layer. Some features are omitted for clarity

POS	…	Prep.	Prep.2	Dist.	Dist.2	Freq.
NN		to		4	0	3
NN			of	2	4	1
NN		of		14	0	49
NN		of	to	2	8	21
NN		of	in	2	6	1
…		...				
NN		between	and	2	2	1
NN		after	of	7	2	1

To simplify the event extraction step and to make the extraction rules generalize well, we combine the extraction rules of the same rule set which obtained from the phrase layer or the clause layer of each event trigger and its specific POS tag to generate a unified rule. For consequent process, each unified rule is represented as a decision table. These decision tables are then used in the extraction step to determine event arguments. Example of such a decision table, which results from the combination of the set of extraction rules in Table 5.1, for the *binding* trigger and its POS tag NN is shown in Table 5.2. In addition, for each decision table of an event trigger, we calculate a confident score based on the frequency of that event trigger being extracted in a layer (e.g. chunk, phrase) divided by frequency of that event trigger being found in the training data.

Table 5.2. A decision table of the *binding* trigger and its POS tag NN used for phrase layer

Feature	Value	Feature	Value
Rule type	*Phrase*	Distance 1	*6*
POS	*NN*	Distance 2	*10*
Trigger	*Binding*	Search order	*right; left; both*
Theme2 score	*0.35*	Preposition	*of; to; in; by*
Confident score	*0.28*	Preposition2	*of:[to, in];*
Theme type	*Protein*	Theme2 type	*Protein*

Since simple, binding and regulatory events have different argument types and there are two types of extraction rules need to be combined (i.e. extraction rules extracted from phrase and clause layers). Therefore, we need 6 variants of the decision tables to represent these unified rules. However, the procedures used to combine these rules are similar in which we use a simple voting scheme based on frequency to select the rules that are more likely reliable and remove those contradict to the selected one. An example of such rule combination algorithms is show in Algorithm 5.2.

Algorithm 5.2. Combining extraction rules of the regulatory events obtained from the clause layer

Input: List of extraction rules **R**, list of event trigger **E**.

Output: Decision tables for each event trigger

```
1 For $e \in \mathbf{E}$
2   Retrieve rules L of e in R
3   Split L into two groups G1, and G2 based on passive feature
4   Sort G1 and G2 based on frequency
5   Select active direction $dr_1$, $dr_2$ for G1 and G2.
6   For i=1 to 2
7     For $g \in \mathbf{G}_i$
8       Remove $r \in g$ if $r_{active\ direction} \neq dr_{i\ active\ direction}$
9       Set distance d1, d2 for $\mathbf{G}_i$
10      Calculate ThemPro, ThemeEvent, CausePro,CauseEvent for $\mathbf{G}_i$
```

5.2.4 Event extraction

In this section we present an algorithm used to extract events from an input sentence. First, the sentence is preprocessed to detect candidate event triggers and to convert into structured representations. Candidate event triggers are detected using the same dictionary as in the learning rules step. Next, we evaluate each candidate trigger and determine its arguments by using a decision table. For each candidate trigger, its decision table is retrieved using a key which consists of the candidate trigger, its POS, the layer containing it, and its event type. For example, if the *binding* trigger belongs to a phrase then its retrieval key is: *binding_NN_Phrase _Binding.*

We model the process of extracting events from a sentence as a pairing task where each protein might be paired with a trigger on the left or on the right of that protein based on a given rule. For example, consider the sentence in Figure 1b: *RFLAT-1activates RANTES gene expression*, with the flat representation it is not clear whether *RANTES* should pair with the *activates* trigger or with the *expression* trigger. However, when mapped into structured representation as show in Figure 5.2, the decision can easily be made based on the syntactic layer. In this case, pairing *RANTES* with the *expression* trigger should have higher priority than pairing with the *activates* trigger. Based on the structured representation, we define an event extraction procedure which consists of three steps: extract events from chunk, from phrase, and from clause. At each layer, the extracted events can be used by its direct upper layer as arguments. The event extraction algorithm is shown in Algorithm 5.3.

Algorithm 5.3. Extracting biomedical events from a sentence

Input: Sentence **S**, a list of proteins **P**, a list of extraction rules **R**, a dictionary **D**, a list of candidate triggers **G**.

Output: list of extracted events.

Sub functions

isChunk(trigger g, protein p): return **True** if g and p can form an event

queryArgs(layer l, trigger g, rule r): query l for a list of protein/events based on the data of r and g.

formEvent(trigger g, List L1, List L2): form events based on g, L1, and L2

hasCause(trigger g, rule r): return **True** if has Cause/Theme2

inRange(trigger g, rule r, trigger g2): return **True** if events formed by g2 can be used as arguments for g.

```
ChunkExtraction(List G, Structure T) // extract events from chunk layer.
1  For $g \in$ G
2      Retrieve chunk rule r of g in R
3      Locate chunk l $\in$ T containing g
4      List L1 = queryArgs(l, g, r)
5      If $r \neq$ null and isChunk(g, p $\in$ L1)
6          formEvent(g, p, null)
7      Else If r is null and isChunk(g, p$\in$L1)
8          Remove g
PhraseExtraction(Trigger g, List G, Structure T) // extract events from phrase layer
1  Repeat
2    List L1 = null;
3    List L2 = null;
4    g = getNext(G) // get next trigger
5    Retrieve phrase rule r of g in R
6    Locate phase l $\in$ T containing g
7    L1 = queryArgs(l,g,r) // theme
8    If  r $\neq$ null and L1= null
9        g2 = getNext(G)  // get next trigger
10       If inRange(g,r,g2)
11           L1 = PhraseExtraction(g2,G,T);
12   If  L1 $\neq$ null
13       If hasCause(g, r)
14           L2= queryArgs(l,g,r) // cause/ theme2
15       formEvent(g,L1,L2)
16 Until empty(G)
ClauseExtraction(List G, Structure T) // extract events from clause layer
1  For $g \in$ G
2      List L1 = null; // theme
3      List L2 = null; //cause / theme 2
4      Retrieve clause rule r of g in R
5      Locate clause l $\in$ T containing g
6      L1 = queryArgs(l, g, r)
7      If $r \neq$ null and L1 $\neq$ null
8        If hasCause(g, r)
9            L2= queryArgs(l,g,r) // cause/ theme2
10     formEvent(g, L1, L2)
Main function
1  Convert s into structured representation T
2  Map p$\in$ P into chunks $\in$ T
3  ChunkExtraction(G,T)
4  PhraseExtraction(null,G,T)
5  ClauseExtraction(G,T)
```

5.2.5 Text preprocessing

The text preprocessing step consist of splitting sentences, replacing protein names with place-holders, tokenizing words into tokens, POS tagging, parsing sentences with a shallow parser, and converting the output of the parser into structured representations. To split the input text (e.g. title, abstract, paragraph) into single sentences, we use the LingPipe sentence splitter (http://alias-i.com/lingpipe/). Sentences that do not contain protein names are skipped. We replace protein/gene names with a place-holder e.g. PRO_i (i is the index of the i^{th} protein/gene in the text) to prevent the parser from segmenting multiple-word protein names and for subsequent processing. Each sentence is then tokenized and tagged with the LingPipe POS tagger. Finally, these outputs (tokens and their POS tags) are parsed with the OpenNLP shallow parser (http://incubator.apache.org/opennlp/) to produce chunks.

Converting chunks into structured representation

We adapt the method described in Segura-Bedmar et al. [62] to convert chunks into structured representations. Here complex clauses are split into multiple simple clauses. To reduce the variants of the structured representations, coordinative noun chunks are merged into one noun chunk as mentioned in section 5.2.1. Furthermore, if an adjective chunk that immediately follows a verb chunk, we merge that adjective chunk into the verb chunk. For example, the following chunks: [VB is] [ADJ dependent] are merged into [VB is dependent], the merged chunk is considered as a verb chunk.

5.3 Results and discussion

5.3.1 Datasets

We use the Genia Event Extraction datasets provided by the BioNLP'11 (https://sites.google.com/site/bionlpst/home/genia-event-extraction-genia) to evaluate our extraction method. The datasets consist of training, development, and test datasets. For training and development datasets, a list of proteins/genes and annotated events are given. For the test set, only a list of proteins/genes is given. Statistics of the datasets are shown in Table 5.3.

Table 5.3. Statistics of training, development, and test datasets. Values in parentheses denote the number of full papers

Items	Training	Development	Test
Abstracts + full papers	800 (+5)	150 (+5)	260 (+4)
Sentences	8759	2954	3437
Proteins	11625	4690	5301
Events	10310	3250	4457
Availability of events	Yes	Yes	No

5.3.2 Evaluation settings

We use both training and development datasets for building the dictionary and learning extraction rules, which are then used to extract biomedical events from the test dataset (Task 1). The extracted events are submitted to the online evaluation system (https://sites.google.com/site/bionlpst/home) to evaluate the results. To obtain the realistic results, we do not apply any tuning techniques to optimize for F-score of the test dataset. We set the thresholds for dictionary entries and confidence scores of the extraction rules to 0.1 and 0.03, respectively. These threshold values are determined empirically based on the development dataset.

5.3.3 Event extraction

Table 5.4 shows the results of our extraction method evaluated on the test dataset using the *Approximate Span/Approximate Recursive matching* criteria. We present the evaluation results of the abstract and full text datasets in parallel to easy analyze the results of both types of text. The data show that our system performs well on both abstract and full text datasets. In particular, it achieves the best results on simple events (SVT-TOTAL), followed by binding events and regulatory events. Overall, the results on the full text dataset are better than on the abstract data set. The results on simple and binding events in the full text are significantly better than in the abstract dataset with 8-10 F-score points higher, whereas the results on the regulatory events in the abstract dataset are slightly better than in the full text dataset.

Table 5.4. Evaluation results of the test dataset. R, P, and F denote recall, precision and F-score, respectively

Event Class	Abstract			Full text		
	R	P	F	R	P	F
Gene_Expression	63.43	85.29	72.76	76.43	88.43	81.99
Transcription	58.39	81.63	68.09	40.54	100.00	57.69
Pro_catabolism	42.86	60.00	50.00	100.00	100.00	100.00
Phosphorylation	71.11	96.00	81.70	88.00	95.65	91.67
Localization	47.70	97.65	64.09	64.71	55.00	59.46
SVT-TOTAL	61.17	87.11	71.87	**74.03**	**87.96**	**80.39**
Binding	39.48	64.93	49.10	**56.25**	**65.32**	**60.45**
EVT-TOTAL	56.25	82.61	66.93	69.19	81.70	74.92
Regulation	20.62	52.17	29.56	17.02	42.11	24.24
Pos_regulation	32.55	56.44	41.29	29.57	51.13	37.47
Neg_regulation	24.54	49.21	32.75	25.52	49.49	33.68
REG-TOTAL	**28.61**	**54.31**	**37.48**	26.94	49.88	34.99
ALL-TOTAL	41.89	**69.72**	52.34	44.47	66.63	**53.34**

Table 5.5 and Table 5.6 present comparison results of our system and the top four systems that participated in the BioNLP’11 challenge. To study the results of each system in more detail we also present the evaluation results on each dataset separately, where Table 5.5 show the results on the abstract dataset and Table 5.6 show the results on the full text dataset.

Table 5.5 shows that our system achieves good results compared to the best system on simple (SVT) and binding (BIND) events. In fact, it performs slightly better than that system on simple events. However, the performance on regulatory (REG) events is lower than best system, with a gap of 12 F-score points. Overall, on the abstract dataset, the Riedel and Andrew [133] system is the best since their system yields the highest F-score, whereas our system yields good results in terms of precision and the Björne and Salakoski [102] system achieves good results in terms of recall. Furthermore, the abstract dataset is the same dataset as used in the BioNLP’09, indicating that the performance of the current extraction methods has been improved, with the best system 4 F-score points higher than the previous state-of-the-art system (56.05% vs. 51.95%) [129].

Table 5.5. Performance comparison on the abstract dataset

System	SVT	BIND	REG	TOTAL		
	F	F	F	R	P	F
Riedel [101]	71.54	**50.76**	**45.51**	48.74	65.94	**56.05**
Björne [102]	70.36	47.50	44.30	**50.06**	59.48	54.37
Quirk [103]	70.08	43.86	40.85	48.52	56.47	52.20
Kilicoglu [64]	67.75	37.41	40.96	43.09	60.37	50.28
Ours	**71.87**	49.10	37.48	41.89	**69.72**	52.34

Table 5.6. Performance comparison on the full text dataset

System	SVT	BIND	REG	TOTAL		
	F	F	F	R	P	F
Riedel [101]	79.18	44.44	**40.12**	47.84	59.76	53.14
Björne [102]	76.98	34.39	39.16	**48.31**	53.38	50.72
Kilicoglu [64]	78.39	35.56	38.12	44.71	57.75	50.40
Quirk [103]	75.63	36.14	38.21	48.94	50.77	49.84
Ours	**80.39**	**60.45**	34.99	44.47	**66.63**	**53.34**

With the inclusion of full text documents in the test dataset, the BioNLP’11 brings more challenges to the event extraction task than the BioNLP’09 does. This requires the adaption ability of each system since the structure and content of biomedical abstracts and full text bodies are different [139]. The results in Table 5.6 show that all systems yield better F-scores on the simple events in the full text dataset than in the abstract dataset. Our system still leads on this type of events. Interestingly, while the other systems drop the performance on the binding events, our system gains 11 F-score points, from 49.11% for the abstract dataset to 60.45% for this dataset. This yields the best result on binding events reported so far. A closer look at the results on the binding events of these systems (data not shown) showed that while two rule-based systems achieve nearly the same precision on the binding events in both datasets (e.g. 49.76% vs. 49.38% for Kilicoglu’s system; 64.93% vs. 65.34% for our system), three ML-based systems drop precision significantly (e.g. 60.89% vs. 47.62 for Riedel’s system; 50% vs. 31.76% for Björne’s system; 44.51%

vs. 32.77% for Quirk's system). This might be due to over-fitting since the number of binding events which is available for training in the abstract dataset is much higher than in the full text dataset (881 events vs. 101 events). This implies that, for binding events, the aforementioned rule-based systems generalize better than their counterpart ML-based systems. This finding was also pointed out by Kilicoglu and Bergler [64]. For regulatory events, our system drops the performance on this dataset but the gap of results between our system and the best system on these event classes is reduced to 6 F-score points. Overall, on full text dataset our system outperforms the best system of the BioNLP'11 in terms of both precision and F-score.

5.3.4 Performance time

When the system is applied to large-scale extractions such as the whole PubMed database or used in Question-Answer systems as envisioned by Wren [140], then computational resources required to run the system should be taken into account. Despite this obvious fact, only few systems report on the computational time needed to run their systems. Riedel & McCallum [133] report that their system needs from 60ms to 297ms, depending on the learning models, to extract events from a sentence. However, this is not included the parsing times and features extraction times. Björne et al. [118] report in their large-scale experiment that, on average, their system needs 954ms to parse a sentence and 486ms to extract events from that sentence. Since Riedel & McCallum [133] use the same parser as Björne et al. [118], we assume that the parsing times of the two systems are equal. Therefore, both systems need from 1040ms to 1400ms to extract events from a sentence. In contrast, our system needs 6.4ms to do so. Details of the performance times are shown in Table 5.7.

Table 5.7. Performance times of our system on the test set and training set. The experiments are run on a PC with Core™i5 2.3MHz CPU, 4 GB of memory

Dataset	# sentences processed	Text prepossessing (average)	Event extraction (average)
Test set	2067	5.9ms	0.49ms
Training set	5186	4.9ms	0.68ms

In general, it is not straightforward to directly compare the computational times between systems due to the differences in hardware as well as other factors e.g. the length of sentences and the number of events per sentence. These differences are shown in Table 5.7 where the text processing times and event extraction times vary on two datasets even though they are run on the same system. However, it is apparent that our system outperforms the mentioned systems, being a 150-fold faster in terms of computational times.

5.3.5 Performance analysis

The results in the previous sections show that our event extraction system outperforms the existing rule-based systems in terms of both precision and recall. In general, its recall is lower than the recall of ML-based systems but the precision of our system is higher than these systems. Furthermore, our system generalizes well on full text dataset and is computationally effective. These characteristics are very important since full text documents are potentially provide richer source of information for events to be extracted but also require more computational resources.

There are some issues which affect our system performance, especially in terms of recall. In the following section we address these issues and discuss possible directions to improve the overall performance.

First, most event extraction systems use a dictionary to detect candidate event triggers however the construction and the use of such a dictionary vary between systems [121]. We also use a dictionary to detect candidate event triggers from input text. Each entry in our dictionary is assigned a confident score as proposed by Kilicoglu and Bergler [127]. In our experiments, we set the threshold for confident score of the dictionary entries to 0.1. We found that raising the threshold would increase precision for some event classes but this would also decrease recall of the others. Furthermore, our extraction algorithm relies on both candidate triggers and proteins to extract events. If some candidate triggers are filtered by the threshold then the procedure used to find event arguments fail to return a desired proteins/events list. In addition, a recent comprehensive study carried out by Tikk et al. [33] on extraction methods for protein-protein interactions reveals that systems are tuned using specific-corpus parameters drop F-score considerably when evaluated on unknown characteristic datasets. Therefore we decided not to tune the dictionary threshold for performance.

Table 5.8 shows recall of the dictionary (see column 3) evaluated on the training and development datasets. The results indicate that the recall varies significantly among event classes where binding and regulatory events have lower recall than the simple events. This is one of the reasons why the performance of the binding and regulatory events is lower than the simple events. Furthermore, the effect of the dictionary on the learning phase can be observed clearly in Table 5.8. In which, the percentage of events learned from each event class is affected by the recall of the dictionary for that event class. This value can be considered as the performance upper bound (PUB) of that event class and it is observed that the higher PUB of an event class, the better performance that class may achieve. For instance, among three event classes: Gene_Expr, Binding, and Pos.Reg, the Gene_Expr class has the highest PUB therefore it achieves good F-score (89 vs. 72) whereas the Binding class has lower PUB and also achieves lower F-score (77 vs. 49) and the Positive_Regulation class achieves the lowest F-score due to its lowest PUB (70 vs. 41.29).

Table 5.8. Statistics of the learning phase on the training and development datasets

Event type	# events	Recall of dict.	% event learned	# unified rules
Gene_Expession	3014	91	89	70
Transcription	825	72	72	20
Pro_catabolism	133	87	87	6
Phosphorylation	303	93	89	8
Localization	348	81	79	29
Binding	1363	79	77	72
Regulation	1409	77	68	98
Pos_Regulation	4385	78	70	220
Neg_Regulation	1780	76	65	143

The second factor that affects the system performance is the rule combination step. While combining extraction rules definitely simplifies the event extraction method, it also causes the loss of information. As shown in the Algorithm 5.3 (line 8), during the combination process, we remove some rules that contradict to the selected rules. This leads to lower recall of the system. Furthermore, the loss of information is clearly visible in the case of binding and regulatory events. For example, the *CausePro*, and *CauseEvent* values in the decision table of an event trigger can provide statistical data for that event trigger such as indicating how often that trigger might have a *cause* as a protein or as an event. However, these data do not tell in which particular case that trigger has a *cause*. When analyzing 50 false positive regulatory events obtained from the developing dataset, we found that of 22 cases due to wrong event classes (e.g. Positive_Regulation vs. Regulation) and of 28 cases due to wrong number of arguments (e.g. with or without causes) or wrong argument types (e.g. protein vs. event). Therefore, to reduce the false positive regulatory events, we need a better strategy such as adding more specific features for these event classes or modify the current extraction algorithm to retain more rules.

Finally, the variants of event triggers in each event class also affect the system performance. Table 5.8 shows that the variants in regulatory classes, which indicated by the number of unified rules, are much higher than in the simple classes. In our experiments, we set the threshold for all unified rules to 0.03. We found that setting threshold for each event class or even individual trigger could increase recall for some cases. However, as we mentioned, this is specific-corpus parameter tuning, we decided not doing so.

5.4 Conclusion

In this chapter we have proposed a novel rule-based method to extract biomedical events from text. Our core method to event extraction is the use of a structured representation to decompose nested and complex events into syntactic layers. This representation not only allows us simplifying the learning and extraction phases but also requires less syntactic analysis of input sentences. The evaluation results show that our system performs well on both abstract and full text datasets. Furthermore, it achieves superior performance in terms of speed. It is clearly suited for large-scale experiments.

Our event extraction method is simpler than the existing ML-based approaches. It is also more robust than the previous proposed rule-based approaches since it learns rules automatically from training data. Its simplicity and robustness has been proven by the performance on simple and binding events. Our approach still has modest performance on regulatory events however this issue has been addressed and therefore opens opportunities for improvements. Its structured representation is generic and is capable to represent any relation types. The proposed feature sets based on the structured representation consist of mainly generic features and a few specific features. Therefore it is suited to extract many types of relations. If needed, its specific features can be easily adapted to any new domain.

Chapter 6. Discussion and Conclusion

Relation extraction methods for biomedical texts play a crucial role in automatically gathering facts and evidence necessary for life sciences. Although many relation extraction methods have been proposed, this task remains challenging due to many factors ranging from the inherent complexity of the natural language, many types of relations need to be extracted, to the lack of training data and suitable techniques. To improve the performance of relation extraction methods, many aspects need to be further studied such as the characteristics of different relation types, the use of NLP tools in relation extraction tasks, the selection of extraction techniques, and the availability of training data. Understanding these aspects may result in a better design of relation extraction methods. As an attempt to contribute to this research field, in this thesis we have investigated three relation extraction tasks with various settings such as relation types and the availability of training data. By studying methods to extract these relation types, we can understand their common characteristics and requirements. This helps designing extraction methods that can generalize well for the other relation types. By varying the use of NLP tools and the availability of training data, we can understand how NLP tools and training data influence the performance of extraction techniques. In particular, these understandings help us to answer the research questions raised in Chapter 1.

6.1 The use of syntactic information for different relation extraction tasks

Syntactic information is widely used for relation extraction tasks. However, the level of syntactic information used in each system varies from simple POS tags to complex structures such as dependency parse trees. Recent methods have a tendency to intensively use the combination of many types of syntactic information which results from NLP tools. This is based on the hypothesis that these types of information can complement each other and eventually boost the performance of the relation systems. Although this strategy shows an increase in the performance of the systems where the training dataset and test dataset have similar characteristics, their performance deteriorates significantly when evaluated on the other test datasets with slightly different characteristics [33]. One of the reasons for this performance degradation might be that the more syntactic information is used the deeper the dependency of the systems to the training data. It is still unclear which types of syntactic information are best suited for the relation extraction tasks.

In this thesis, we use syntactic information to extract three relation types. We have observed two properties. First, in most cases, the relations between two entities can be expressed in two abstract forms: subject-(verb)-object and noun phrase. Second, the co-ordination structure is important to determine the boundaries of relations. With respect to the availability of the training data, our relation extraction approaches change from manually defined rules to automatically learning rules from training data. Interestingly, the level of syntactic analysis required in each approach is gradually reduced while the complexity of extraction tasks is increased. In particular, for extracting causal relations on HIV drug resistance, due to unavailable training data, we use rule-based approach which relies on grammatical relations derived from the syntactic parse trees. In case of extracting protein-protein interactions, the training data are available but without relation words describing the interactions between protein pairs. We use three syntactic patterns which represent the subject-object and noun phrase forms to extract sub-trees from syntactic parse trees. We then apply rules to extract PPIs from the obtained sub-trees. In case of extracting biological events, the training data are fully available. With the observation made from the PPI extraction task that the syntactic information required for relation extraction task can be further reduced to the phrasal structures (e.g. noun phrase, verb phrase, preposition phrase), we use a method that automatically learns rules from training data while only requiring input text to be analyzed by shallow parsing.

In conclusion, we have demonstrated that there are common syntactic patterns between relation extraction tasks in which most of relations can be expressed in subject-object and noun phrase forms. In our study, the use of phrasal structures is sufficient to the relation extraction tasks. The main benefit of full parsing is that it provides an easy way to obtain the subject-object relations and co-ordination structures. However, these structures can also be obtained from shallow parsing by using a small set of rules. Since the extraction of HIV drug resistance and PPI tasks can be considered as the sub-tasks of the biomedical event extraction task, it is reasonable to assume that the method developed for extracting biomedical events can potentially perform well on the first two relation extraction tasks.

6.2 The role of machine learning to relation extraction tasks

With the availability of training data and the increasing complexity of relation tasks, for example the datasets of the BioNLP’11 Shared Task consist of more than 600 event triggers and 13,560 events [60], it is impractical to manually define rules that can cover all extraction cases. Therefore, the use of ML methods to extract relations is obvious and recently has become the main technique to relation extraction tasks. Due to the enormous variants of words used to express the relations, directly use sequence of words surrounding the relations (i.e. flat structure) as the features (lexical features) for ML extraction methods obtains limited results. This problem has been addressed by a study [100] on the linguistic and syntactic characteristics of existing feature sets for PPI extraction tasks which indicate that the advantage of adding lexical features to ML methods gives no significant improvement. Therefore, in many ML relation extraction approaches, the unstructured texts that express the relations have been transformed into structured representations that can be best learned by ML methods. To achieve a good performance, these ML approaches require complex learning algorithms together with a rich feature set [50, 101]. However, these approaches do not generalize well to unknown characteristics text [33], which is the case when applied to real world applications.

To overcome the highly variant nature of textual data which limits the use of ML methods for relation extraction task, we have proposed the partition strategy which split relations into suitable groups to make data more consistent. The benefits of this strategy are 3-fold. First, by grouping relations that have similar characteristics together, we can select the most appropriate features for each group. Second, consistent data make the ML methods more robust and off-the-shelf ML tools can be used. Third, learning model obtained from each group potentially generalize well when applying to new datasets since they are share the same syntactic pattern as mentioned in the section above. The results in chapter 4 show that our PPI extraction method uses a standard ML method but outperforms the state-of-the-art systems on cross-corpora and cross-learning evaluations. Our bio-event extraction method is simple but generalizes well on both abstract and full text datasets.

In conclusion, our study has shown that the ML methods play an important role in relation extraction tasks. ML methods benefit from partitioning dataset based on syntactic patterns since it can reduce the variants of unstructured text. This reduces the complexity of learning methods and makes them generalize well to new domains.

6.3 The performance time of relation extraction systems

When the system is applied to large scale extraction or is integrated into real-time applications such as question-answer systems, performance times required to run the system need to be taken into account. In general, there are two main factors that affect the performance times of a relation extraction system, namely NLP tools and extraction methods.

For extraction methods based on ML techniques, the performance times depend much on the number of features that are used for learning methods. As the complexity of the extraction tasks increases, many complex representations of the relations are proposed, which also means more features are used. For example, the feature sets of a typical ML-based relation extraction system may consist of 25k+ features [96]. However the more features are used, the more performance times are required. Recent studies of feature selection have demonstrated that performance times required for the PPI extraction task can be reduced by 50% when 60% of the original feature sets are removed [91]. Our approaches split relations bases on their syntactic properties and use specific feature sets for each groups of relations (e.g. noun phrase and subject-object groups), thus reducing the number of features significantly.

The second factor that contributes to the computational times is the use of parsing tools. Recently, full parsing is widely used in most of relation extraction methods. The output of the full parser is converted into a dependency format, which is then used as features for ML-based relation extraction systems. In general, the times spent to fully parse a sentence account for more than 70% of total times needed for the system to extract relations from a sentence [118]. In our method to extract biomedical events, we only use a shallow parser to analyze input sentences, which is less computational resources demanding compared to that of full parsing. Overall, its performance in terms of run-time is 150-fold faster than the state-of-the-art systems.

In conclusion, our study demonstrates that performance times required for relation extraction tasks can be reduced significantly by partitioning data and using a suitable level of syntactic analysis.

6.4 Contribution

The main contributions of this thesis are the methods that we proposed to extraction relations and the enhancement to the existing relation extraction tasks in terms of precision and performance time. More specific, our contributions through three relation extraction tasks are as follows:

We introduce a novel method to extract causal relations on HIV drug resistance. It is the first method of its kind to extract this type of relations and to combine the extracted relations. The results show that our system achieves good performance when compared with the expert systems. Our system is being deployed in the ViroLab project (www.ViroLab.org) to help virologists find evidence for HIV drug resistance from literature in a controlled and automated way.

For the protein-protein interactions (PPIs) extraction task, we propose a hybrid system that employs both rule- and ML-based method. By introducing a data partition strategy, we can significantly reduce the number of features used by the ML classifier which then increases the robustness of the system. We demonstrate that our system achieves a performance that is comparable with the state-of-the-art systems when evaluated using 10-fold cross validation. However, it outperforms these systems when evaluated using cross-corpora criteria, in which, training data and testing data might have completely different properties, meaning that this setting is closer to the real world situation. Furthermore, our system also achieves the best performance in terms of speed.

We present a novel rule-based method to extract biomedical events from texts which consists of two phases: a learning phase and an extraction phase. By using a structured representation, we can decompose the complex and nested structures of biomedical events into simple syntactic layers. Based on this structure, the system both learns rules to extract events from training data and applies rules to extract events from text. This representation not only allows us to simplify the learning and extraction phases but it also requires less syntactic analysis of input sentences. The evaluation results show that our system performs well on both abstract and full text datasets. Furthermore, it achieves superior performance in terms of speed, ranging from 150 to 200-fold faster than the state-of-the-art systems. It is clearly suited for large-scale experiments. In addition, our approach is simple and generic; it can easily be adapted to extract any types of relations.

6.5 Future work

In addition to the relation extraction methods that we have proposed, our study opens up several opportunities for future work.

1. The use of shallow parsing in our event extraction method has shown promising results. However, the output from the shallow parser needs to convert into structured representations in order to use for relation extraction method. In our work, we use a set of simple rules to do this conversion task. As a consequence, there are many syntactic variants we have not studies and taken into account. We expect that a better implementation of conversion tool will improve the performance of our current system. Implementation such tool is much easier and faster compared to that of full parsing.

2. In our method to event extraction, we use decision tables to determine arguments and argument types for binding and regulatory events. However, the algorithm used to form these decision tables causes the loss of information, which leads to the degradation of recall for these event types significantly. Therefore, a better method to determine arguments and their types are needed to improve the system performance. Furthermore, instead of using the same feature set for all event types, we can define three different feature sets for simple, binding, and regulatory events. This would capture better properties of binding and regulatory events, which eventually boosts the overall performance.

3. For extracting causal relations on HIV drug resistance, some improvements can be carried out. First, we can use the event extraction method to extract causal relations. Since this method is robust, we can apply it to full text

documents to obtain larger set of relation pairs. Second, we define a new data structure that can retain the original context of the extracted relation pairs, thus avoiding the loss of meaning compared to the current binary representation. Finally, a bootstrapping method can be used to expand the annotated dataset required by the learning method.

References

1. Ananiadou S, Pyysalo S, Tsujii J, Kell DB: **Event extraction for systems biology by text mining the literature.** *Trends in biotechnology* 2010, **28**:381-90.

2. Faro A, Giordano D, Spampinato C: **Combining literature text mining with microarray data: advances for system biology modeling.** *Briefings in bioinformatics* 2011, **13**:61-82.

3. Jensen LJ, Saric J, Bork P: **Literature mining for the biologist : from information retrieval to biological discovery**. *Nature Review* 2006, **7**:119-129.

4. Andronis C, Sharma A, Virvilis V, Deftereos S, Persidis A: **Literature mining, ontologies and information visualization for drug repurposing.** *Briefings in bioinformatics* 2011, **12**.

5. Garten Y, Coulet A, Altman RB: **Recent progress in automatically extracting information from the pharmacogenomic literature.** *Pharmacogenomics* 2010, **11**:1467-89.

6. Ananiadou S, Kell DB, Tsujii J-ichi: **Text mining and its potential applications in systems biology.** *Trends in biotechnology* 2006, **24**:571-9.

7. Chapman WW, Cohen KB: **Current issues in biomedical text mining and natural language processing.** *Journal of biomedical informatics* 2009, **42**:757-9.

8. Zweigenbaum P, Demner-Fushman D, Yu H, Cohen KB: **Frontiers of biomedical text mining: current progress.** *Briefings in bioinformatics* 2007, **8**:358-75.

9. Riedman CAF: **Automated Acquisition of Disease – Drug Knowledge from Biomedical and Clinical Documents : An Initial Study**. *Journal of the American Medical Informatics Association* 2008, **15**:87-98.

10. Shetty KD, Dalal SR: **Using information mining of the medical literature to improve drug safety.** *Journal of the American Medical Informatics Association : JAMIA* 2011:1-7.

11. Bankhead A, Mancini E, Sims AC, Baric RS, McWeeney S, Sloot PMA: **A simulation framework to investigate in vitro viral infection dynamics**. *Journal of Computational Science* 2011.

12. Sloot PMA, Coveney PV, Ertaylan G, Müller V, Boucher CA, Bubak M: **HIV decision support: from molecule to man.** *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences* 2009, **367**:2691-703.

13. van Dijk D, Ertaylan G, Boucher CA, Sloot PMA: **Identifying potential survival strategies of HIV-1 through virus-host protein interaction networks.** *BMC systems biology* 2010, **4**:96.

14. Mei S, Quax R, van de Vijver D, Zhu Y, Sloot PMA: **Increasing risk behaviour can outweigh the benefits of antiretroviral drug treatment on the HIV incidence among men-having-sex-with-men in Amsterdam.** *BMC infectious diseases* 2011, **11**:118.

15. Senger C, Grüning BA, Erxleben A, Döring K, Patel H, Flemming S, Merfort I, Günther S: **Mining and Evaluation of Molecular Relationships in Literature.** *Bioinformatics* 2012, **28**:709-714.

16. Wang J, Zhang Y, Marian C, Ressom HW: **Identification of aberrant pathways and network activities from high-throughput data.** *Briefings in bioinformatics* 2012.

17. Jelier R, Goeman JJ, Hettne KM, Schuemie MJ, den Dunnen JT,'t Hoen PAC: **Literature-aided interpretation of gene expression data with the weighted global test.** *Briefings in bioinformatics* 2011.

18. Tsuruoka Y, Miwa M, Hamamoto K, Tsujii J, Ananiadou S: **Discovering and visualizing indirect associations between biomedical concepts.** *Bioinformatics* 2011, **27**:i111-i119.

19. Mons B, van Haagen H, Chichester C, Hoen P-B 't, den Dunnen JT, van Ommen G, van Mulligen E, Singh B, Hooft R, Roos M, Hammond J, Kiesel B, Giardine B, Velterop J, Groth P, Schultes E: **The value of data.** *Nature genetics* 2011, **43**:281-3.

20. Leitner F, Valencia A: **A text-mining perspective on the requirements for electronically annotated abstracts**. *FEBS Letters* 2008, **582**:1178-1181.

21. Bader GD: **BIND: the Biomolecular Interaction Network Database**. *Nucleic Acids Research* 2003, **31**:248-250.

22. Reguly T, Breitkreutz A, Boucher L, Breitkreutz B-joe, Hon GC, Myers CL, Parsons A, Friesen H, Oughtred R, Tong A, Stark C, Ho Y, Botstein D, Andrews B, Boone C, Troyanskya OG, Ideker T, Dolinski K, Batada NN, Tyers M: **Comprehensive curation and analysis of global interaction networks in Saccharomyces cerevisiae.** *Journal of biology* 2006, **5**:11.

23. Mishra GR, Suresh M, Kumaran K, Kannabiran N, Suresh S, Bala P, Shivakumar K, Anuradha N, Reddy R, Raghavan TM, Menon S, Hanumanthu G, Gupta M, Upendran S, Gupta S, Mahesh M, Jacob B, Mathew P, Chatterjee P, Arun KS, Sharma S, Chandrika KN, Deshpande N, Palvankar K, Raghavnath R, Krishnakanth R, Karathia H, Rekha B, Nayak R, Vishnupriya G, Kumar HGM, Nagini M, Kumar GSS, Jose R, Deepthi P, Mohan SS, Gandhi TKB, Harsha HC, Deshpande KS, Sarker M, Prasad TSK, Pandey A: **Human protein reference database--2006 update.** *Nucleic acids research* 2006, **34**:D411-4.

24. Cusick ME, Yu H, Smolyar A, Venkatesan K, Carvunis A-ruxandra, Simonis N, Rual J-françois, Borick H, Braun P, Dreze M, Vandenhaute J, Galli M, Yazaki J, Hill DE, Ecker JR, Roth FP, Vidal M: **Literature-curated protein interaction datasets perspective**. *Nature Methods* 2009, **6**:39-46.

25. Ananiadou SM, John: *Text Mining for Biology and Biomedicine*. ARTECH HOUSE; 2006, **33**:300.

26. Miyao Y, Sagae K, Saetre R, Matsuzaki T, Tsujii J: **Evaluating contributions of natural language parsers to protein-protein interaction extraction.** *Bioinformatics* 2009, **25**:394-400.

27. Kang N, van Mulligen EM, Kors JA: **Comparing and combining chunkers of biomedical text.** *Journal of biomedical informatics* 2011, **44**:354-60.

28. Zhou D, He Y: **Extracting interactions between proteins from the literature**. *Journal of Biomedical Informatics* 2008, **41**:393-407.

29. Pyysalo S, Airola A, Heimonen J, Björne J, Ginter F, Salakoski T: **Comparative analysis of five protein-protein interaction corpora.** *BMC bioinformatics* 2008, **9 Suppl 3**:S6.

30. Jaeger S, Gaudan S, Leser U, Rebholz-Schuhmann D: **Integrating protein-protein interactions and text mining for protein function prediction.** *BMC bioinformatics* 2008, **9 Suppl 8**:S2.

31. Oda K, Kim JD, Ohta T, Okanohara D, Matsuzaki T, Tateisi Y, Tsujii J: **New challenges for text mining: mapping between text and manually curated pathways.** *BMC bioinformatics* 2008, **9 Suppl 3**:S5.

32. Kilicoglu H, Rosemblat G, Fiszman M, Rindflesch TC: **Constructing a semantic predication gold standard from the biomedical literature**. *BMC Bioinformatics* 2011, **12**:486.

33. Tikk D, Thomas P, Palaga P, Hakenberg J, Leser U: **A comprehensive benchmark of kernel methods to extract protein-protein interactions from literature.** *PLoS computational biology* 2010, **6**:e1000837.

34. Kabiljo R, Clegg AB, Shepherd AJ: **A realistic assessment of methods for extracting gene/protein interactions from free text.** *BMC bioinformatics* 2009, **10**:233.

35. Katrenko S: **A Closer Look At Learning Relations From Text**. PhD Thesis 2009:241.

36. Hakenberg J: **Mining Relations from the Biomedical Literature**. PhD Thesis 2009:179.

37. Fayruzov T: **Mining and Modelling Interaction Networks for Systems Biology**. PhD Thesis 2010:204.

38. Erhardt R a-a, Schneider R, Blaschke C: **Status of text-mining techniques applied to biomedical text.** *Drug discovery today* 2006, **11**:315-25.

39. Ohta T, Pyysalo S, Kim J-D, Tsujii J: **A Re-Evaluation of Biomedical Named Entity–Term Relations**. *Journal of Bioinformatics and Computational Biology* 2010, **08**:917.

40. Leaman R, Gonzalez G: **BANNER: an executable survey of advances in biomedical named entity recognition.** *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing* 2008, **663**:652-63.

41. Hawizy L, Jessop DM, Adams N, Murray-Rust P: **ChemicalTagger: A tool for semantic text-mining in chemistry.** *Journal of cheminformatics* 2011, **3**:17.

42. Sasaki Y, Tsuruoka Y, McNaught J, Ananiadou S: **How to make the most of NE dictionaries in statistical NER.** *BMC bioinformatics* 2008, **9 Suppl 11**:S5.

43. Giles CB, Wren JD: **Large-scale directional relationship extraction and resolution.** *BMC bioinformatics* 2008, **9 Suppl 9**:S11.

44. Barrett N, Weber-Jahnke J: **Building a biomedical tokenizer using the token lattice design pattern and the adapted Viterbi algorithm.** *BMC bioinformatics* 2011, **12 Suppl 3**:S1.

45. Saetre R, Yoshida K, Miwa M, Matsuzaki T, Kano Y, Tsujii J: **Extracting protein interactions from text with the unified AkaneRE event extraction system.** *IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM* 2010, **7**:442-53.

46. Smith L, Rindflesch T, Wilbur WJ: **MedPost: a part-of-speech tagger for bioMedical text.** *Bioinformatics* 2004, **20**:2320-1.

47. Tsuruoka Y, Tateisi Y, Kim JD, Ohta T, McNaught J, Ananiadou S, Tsujii J: **Developing a Robust Part-of-Speech Tagger for Biomedical Text**. In *Advances in Informatics - 10th Panhellenic Conference on Informatics*. 2005, **3746**.

48. Rinaldi F, Schneider G, Kaljurand K, Hess M, Andronis C, Konstandi O, Persidis A: **Mining of relations between proteins over biomedical scientific literature using a deep-linguistic approach.** *Artificial intelligence in medicine* 2007, **39**:127-36.

49. Witten IH, Frank E: *Data Mining: Practical Machine Learning Tools and Techniques*. 2nd edition. Morgan Kaufmann Publishers; 2005:525.

50. Miwa M, Saetre R, Miyao Y, Tsujii J: **Protein-protein interaction extraction by leveraging multiple kernels and parsers.** *International journal of medical informatics* 2009, **78**:e39-46.

51. Kim S, Yoon J, Yang J: **Kernel approaches for genic interaction extraction.** *Bioinformatics* 2008, **24**:118-26.

52. Kim J-D, Ohta T, Tateisi Y, Tsujii J: **GENIA corpus--a semantically annotated corpus for bio-textmining**. *Bioinformatics* 2003, **19**:i180-i182.

53. Bunescu R, Ge R, Kate RJ, Marcotte EM, Mooney RJ, Ramani AK, Wong YW: **Comparative experiments on learning information extractors for proteins and their interactions.** *Artificial intelligence in medicine* 2005, **33**:139-55.

54. Pyysalo S, Ginter F, Heimonen J, Björne J, Boberg J, Järvinen J, Salakoski T: **BioInfer: a corpus for information extraction in the biomedical domain.** *BMC bioinformatics* 2007, **8**:50.

55. Fundel K, Küffner R, Zimmer R: **RelEx--relation extraction using dependency parse trees.** *Bioinformatics* 2007, **23**:365-71.

56. Ding J, Berleant D, Nettleton D, Wurtele E: **Mining medline: abstracts, sentences, or phrases?** In *Pac Symp Biocomput*. 2002:326-337.

57. Nédellec C: **Learning Language in Logic - Genic Interaction Extraction Challenge**. In *The ICML05 workshop: Learning Language in Logic (LLL'05)*. Bonn, Germany: 2005:97-99.

58. Airola A, Pyysalo S, Björne J, Pahikkala T, Ginter F, Salakoski T: **All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning.** *BMC bioinformatics* 2008, **9 Suppl 11**:S2.

59. Kim JD, Ohta T, Tsujii J: **Corpus annotation for mining biomedical events from literature.** *BMC bioinformatics* 2008, **9**:10.

60. Kim JD, Wang Y, Takagi T, Yonezawa A: **Overview of Genia Event Task in BioNLP Shared Task 2011**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:7-15.

61. Huang M, Zhu X, Li M: **A hybrid method for relation extraction from biomedical literature.** *International journal of medical informatics* 2006, **75**:443-55.

62. Segura-Bedmar I, Martínez P, de Pablo-Sánchez C: **A linguistic rule-based approach to extract drug-drug interactions from pharmacological documents.** *BMC bioinformatics* 2011, **12 Suppl 2**:S1.

63. Miwa M, Sætre R, Miyao Y, Tsujii J: **A Rich Feature Vector for Protein-Protein Interaction Extraction from Multiple Corpora**. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics; 2009:121-130.

64. Kilicoglu H, Bergler S: **Adapting a General Semantic Interpretation Approach to Biological Event Extraction**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:173-182.

65. Baumgartner WA, Cohen KB, Hunter L: **An open-source framework for large-scale, flexible evaluation of biomedical text mining systems.** *Journal of biomedical discovery and collaboration* 2008, **3**:1.

66. Leitner F, Mardis SA, Krallinger M, Cesareni G, Hirschman LA, Valencia A: **An Overview of BioCreative II.5.** *IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM* 2010, **7**:385-99.

67. He M, Wang Y, Li W: **PPI finder: a mining tool for human protein-protein interactions.** *PloS one* 2009, **4**:e4554.

68. Torvik VI, Smalheiser NR: **A quantitative model for linking two disparate sets of articles in MEDLINE.** *Bioinformatics* 2007, **23**:1658-65.

69. Zhou D, He Y, Kwoh CK: **From Biomedical Literature to Knowledge : Mining Protein-Protein Interactions**. In *Comp. Intel. in Biomed. & Bioinform*. Springer; 2008:397-421.

70. Hao Y, Zhu X, Huang M, Li M: **Discovering patterns to extract protein-protein interactions from the literature: Part II.** *Bioinformatics* 2005, **21**:3294-300.

71. Yakushiji A: **Automatic Construction of Predicate-argument Structure Patterns for Biomedical Information Extraction**. In *EMNLP '06 Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. 2006:284-292.

72. Wang H-C, Chen Y-H, Kao H-Y, Tsai S-J: **Inference of transcriptional regulatory network by bootstrapping patterns.** *Bioinformatics* 2011, **27**:1422-8.

73. Liu H, Komandur R, Verspoor K: **From Graphs to Events : A Subgraph Matching Approach for Information Extraction from Biomedical Text**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:164-172.

74. Hakenberg J, Leaman R, Vo NH, Jonnalagadda S, Sullivan R, Miller C, Tari L, Baral C, Gonzalez G: **Efficient extraction of protein-protein interactions from full-text articles.** *IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM* 2010, **7**:481-94.

75. Nguyen QL, Tikk D, Leser U: **Simple tricks for improving pattern-based information extraction from the biomedical literature.** *Journal of biomedical semantics* 2010, **1**:9.

76. Rebholz-Schuhmann D, Jimeno-Yepes A, Arregui M, Kirsch H: **Measuring prediction capacity of individual verbs for the identification of protein interactions.** *Journal of biomedical informatics* 2010, **43**:200-7.

77. Fox AD, Jr WAB, Johnson HL, Hunter LE, Slonim DK: **Mining Protein-Protein Interactions from GeneRIFs with OpenDMAP**. In *LNBI6004*. 2010:43-52.

78. Choi YS: **Tree pattern expression for extracting information from syntactically parsed text corpora**. *Data Mining and Knowledge Discovery* 2010, **22**:211-231.

79. Spasic I, Sarafraz F, Keane JA, Nenadic G: **Medication information extraction with linguistic pattern matching and semantic rules.** *Journal of the American Medical Informatics Association : JAMIA* 2010, **17**:532-5.

80. Jang H, Lim J, Lim J-H, Park S-J, Lee K-C, Park S-H: **Finding the evidence for protein-protein interactions from PubMed abstracts.** *Bioinformatics* 2006, **22**:e220-6.

81. Ono T, Hishigaki H, Tanigami A, Takagi T: **Automated extraction of information on protein-protein interactions from the biological literature.** *Bioinformatics* 2001, **17**:155-61.

82. Koike A, Niwa Y, Takagi T: **Automatic extraction of gene/protein biological functions from biomedical text.** *Bioinformatics* 2005, **21**:1227-36.

83. Kim J-H, Mitchell A, Attwood TK, Hilario M: **Learning to extract relations for protein annotation.** *Bioinformatics* 2007, **23**:i256-63.

84. Rinaldi F, Schneider G, Kaljurand K, Hess M, Romacker M: **An environment for relation mining over richly annotated corpora: the case of GENIA.** *BMC bioinformatics* 2006, **7 Suppl 3**:S3.

85. Kim J-J, Zhang Z, Park JC, Ng S-K: **BioContrasts: extracting and exploiting protein-protein contrastive relations from biomedical literature.** *Bioinformatics* 2006, **22**:597-605.

86. Malik R, Franke L, Siebes A: **Combination of text-mining algorithms increases the performance.** *Bioinformatics* 2006, **22**:2151-7.

87. Giuliano C, Lavelli A, Romano L, Sommarive V: **Exploiting Shallow Linguistic Information for Relation Extraction from Biomedical Literature**. In *ACL 2006*. 2006, **18**:401-408.

88. Katrenko S, Adriaans P: **Learning Relations from Biomedical Corpora Using Dependency Tree Levels**. In *In Proceedings of the Fifteenth Dutch-Belgian Conference on Machine Learning (Benelearn)*. 2006.

89. Erkan G, Ozgur A, Radev DR: **Semi-Supervised Classification for Extracting Protein Interaction Sentences using Dependency Parsing**. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 2007:228-237.

90. Sætre R, Sagae K, Tsujii J: **Syntactic features for protein-protein interaction extraction**. In *The 2nd International Symposium on Languages in Biology and Medicine LBM 2007 Short Papers*. 2007.

91. Landeghem SV, Saeys Y, Peer YVD, Baets BD: **Extracting Protein-Protein Interactions from Text using Rich Feature Vectors and Feature Selection**. In *Proceedings of the Third International Symposium on Semantic Mining in Biomedicine 2008*. 2008.

92. Kim M-Y: **Detection of gene interactions based on syntactic relations.** *Journal of biomedicine & biotechnology* 2008, **2008**:371710.

93. Kim S, Shin S-Y, Lee I-H, Kim S-J, Sriram R, Zhang B-T: **PIE: an online prediction system for protein-protein interactions from text.** *Nucleic acids research* 2008, **36**:W411-5.

94. Ahmed ST, Nair R, Patel C, Davulcu H: **BioEve : Bio-Molecular Event Extraction from Text Using Semantic Classification and Dependency Parsing**. In *Proceedings of the Workshop on BioNLP Shared Task - BioNLP ’09*. 2009:99-102.

95. Niu Y, Otasek D, Jurisica I: **Evaluation of linguistic features useful in extraction of interactions from PubMed; application to annotating known, high-throughput and predicted interactions in I2D.** *Bioinformatics* 2010, **26**:111-9.

96. Van Landeghem S, Abeel T, Saeys Y, Van de Peer Y: **Discriminative and informative features for biomolecular text mining with ensemble feature selection.** *Bioinformatics* 2010, **26**:i554-60.

97. Segura-Bedmar I, Martínez P, de Pablo-Sánchez C: **Using a shallow linguistic kernel for drug-drug interaction extraction.** *Journal of biomedical informatics* 2011.

98. Kim S, Yoon J, Yang J, Park S: **Walk-weighted subsequence kernels for protein-protein interaction extraction.** *BMC bioinformatics* 2010, **11**:107.

99. Li J, Zhang Z, Li X, Chen H: **Kernel-Based Learning for Biomedical**. *Journal of the American Society for Information Science* 2008, **59**:756-769.

100. Fayruzov T, De Cock M, Cornelis C, Hoste V: **Linguistic feature analysis for protein interaction extraction.** *BMC bioinformatics* 2009, **10**:374.

101. Riedel S: **Robust Biomedical Event Extraction with Dual Decomposition and Minimal Domain Adaptation**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:46-50.

102. Björne J, Salakoski T: **Generalizing Biomedical Event Extraction**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:183-191.

103. Quirk C, Choudhury P, Gamon M, Vanderwende L: **MSR-NLP Entry in BioNLP Shared Task 2011**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:155-163.

104. UNAIDS: *AIDS epidemic update: December 2006*. 2006.

105. Richman DD, Margolis DM, Delaney M, Greene WC, Hazuda D, Pomerantz RJ: **The challenge of finding a cure for HIV infection.** *Science (New York, N.Y.)* 2009, **323**:1304-7.

106. Vercauteren J, Vandamme A-M: **Algorithms for the interpretation of HIV-1 genotypic drug resistance information.** *Antiviral research* 2006, **71**:335-42.

107. Lengauer T, Sing T: **Bioinformatics-assisted anti-HIV therapy.** *Nature reviews. Microbiology* 2006, **4**:790-7.

108. Saigo H, Uno T, Tsuda K: **Mining complex genotypic features for predicting HIV-1 drug resistance.** *Bioinformatics* 2007, **23**:2455-62.

109. Cohen AM, Hersh WR: **A survey of current work in biomedical text mining.** *Briefings in bioinformatics* 2005, **6**:57-71.

110. Saric J, Jensen LJ, Ouzounova R, Rojas I, Bork P: **Extraction of regulatory gene/protein networks from Medline.** *Bioinformatics* 2006, **22**:645-50.

111. Chowdhary R, Zhang J, Liu JS: **Bayesian inference of protein-protein interactions from biological literature.** *Bioinformatics* 2009, **25**:1536-42.

112. Abulaish M, Dey L: **Biological relation extraction and query answering from MEDLINE abstracts using ontology-based text mining**. *Data & Knowledge Engineering* 2007, **61**:228-262.

113. Klein D, Manning CD: **Accurate unlexicalized parsing**. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - ACL '03*. Morristown, NJ, USA: Association for Computational Linguistics; 2003, **1**:423-430.

114. Horn F, Lau AL, Cohen FE: **Automated extraction of mutation data from the literature: application of MuteXt to G protein-coupled receptors and nuclear hormone receptors.** *Bioinformatics* 2004, **20**:557-68.

115. Sanchez-Graillet O, Poesio M: **Negation of protein-protein interactions: analysis and extraction.** *Bioinformatics* 2007, **23**:i424-32.

116. Torvik VI, Smalheiser NR: **A quantitative model for linking two disparate sets of articles in MEDLINE.** *Bioinformatics* 2007, **23**:1658-65.

117. Liao JG, Chin K-V: **Logistic regression for disease classification using microarray data: model selection in a large p and small n case.** *Bioinformatics* 2007, **23**:1945-51.

118. Björne J, Ginter F, Pyysalo S, Tsujii J, Salakoski T: **Complex event extraction at PubMed scale.** *Bioinformatics* 2010, **26**:i382-90.

119. Sætre R, Miwa M, Yoshida K, Tsujii J: **From protein-protein interaction to molecular event extraction**. In *Proceedings of BioNLP'09 Shared Task Workshop*. Morristown, NJ, USA: Association for Computational Linguistics; 2009:103-106.

120. Yang Z, Lin H, Li Y: **BioPPISVMExtractor: a protein-protein interaction extractor for biomedical literature using SVM and rich feature sets.** *Journal of biomedical informatics* 2010, **43**:88-96.

121. Kim JD, Ohta T, Pyysalo S, Kano Y, Tsujii J: **Overview of BioNLP'09 shared task on event extraction**. In *Proceedings of BioNLP'09 Shared Task Workshop*. Morristown, NJ, USA: Association for Computational Linguistics; 2009:1-9.

122. Krallinger M, Valencia A, Hirschman L: **Linking genes to literature: text mining, information extraction, and retrieval applications for biology.** *Genome biology* 2008, **9 Suppl 2**:S8.

123. Tari L, Anwar S, Liang S, Cai J, Baral C: **Discovering drug-drug interactions: a text-mining and reasoning approach based on properties of drug metabolism.** *Bioinformatics* 2010, **26**:i547-53.

124. Bui QC, Nualláin BO, Boucher CA, Sloot PMA: **Extracting causal relations on HIV drug resistance from literature.** *BMC Bioinformatics* 2010, **11**:101.

125. Bui QC, Sloot PMA: **Extracting biological events from text using simple syntactic patterns**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:143-146.

126. Kaljurand K, Schneider G, Rinaldi F: **UZurich in the BioNLP 2009 shared task**. In *Proceedings of BioNLP'09 Shared Task Workshop*. Morristown, NJ, USA: Association for Computational Linguistics; 2009:28-36.

127. Kilicoglu H, Bergler S: **Syntactic dependency based heuristics for biological event extraction**. In *Proceedings of BioNLP'09 Shared Task Workshop*. Morristown, NJ, USA: Association for Computational Linguistics; 2009:119-127.

128. Buyko E, Faessler E, Wermter J, Hahn U: **Event extraction from trimmed dependency graphs**. In *Proceedings of BioNLP'09 Shared Task Workshop*. Morristown, NJ, USA: Association for Computational Linguistics; 2009:19-27.

129. Björne J, Heimonen J, Ginter F, Airola A, T: **Extracting complex biological events with rich graph-based feature sets**. In *Proceedings of BioNLP'09 Shared Task Workshop*. 2009:10-18.

130. Miwa M, Sætre R, Kim J-D, Tsujii J: **Event Extraction With Complex Event Classification Using Rich Features**. *Journal of Bioinformatics and Computational Biology* 2010, **08**:131.

131. Miwa M, Pyysalo S, Hara T, Tsujii J: **A Comparative Study of Syntactic Parsers for Event Extraction**. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*. Association for Computational Linguistics; 2010:37-45.

132. Vlachos A, Craven M: **Biomedical Event Extraction from Abstracts and Full Papers using Search-based Structured Prediction**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:36-40.

133. Riedel S, Andrew M: **Fast and Robust Joint Models for Biomedical Event Extraction**. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. 2011:1-12.

134. Poon H, Vanderwende L: **Joint Inference for Knowledge Extraction from Biomedical Literature**. In *The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 2010:813-821.

135. Móra G, Farkas R, Szarvas G, Molnár Z: **Exploring ways beyond the simple supervised learning approach for biological event extraction**. In *Proceedings of BioNLP'09 Shared Task Workshop*. Morristown, NJ, USA: Association for Computational Linguistics; 2009:137-140.

136. Neves ML, Carazo JM, Pascual-Montano A: **Extraction of biomedical events using case-based reasoning**. In *Proceedings of BioNLP'09 Shared Task Workshop*. Morristown, NJ, USA: Association for Computational Linguistics; 2009, **1**:68-76.

137. Mcclosky D, Surdeanu M, Manning CD: **Event Extraction as Dependency Parsing for BioNLP 2011**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:41-45.

138. Cohen KB, Verspoor K, Johnson HL, Roeder C, Ogren PV, Jr WAB, White E, Tipney H, Hunter L: **High-precision biological event extraction with a concept recognizer**. In *Proceedings of BioNLP'09 Shared Task Workshop*. 2009:50-58.

139. Cohen KB, Johnson HL, Verspoor K, Roeder C, Hunter LE: **The structural and content aspects of abstracts versus bodies of full text journal articles are different.** *BMC bioinformatics* 2010, **11**:492.

140. Wren JD: **Question answering systems in biology and medicine--the time is now.** *Bioinformatics* 2011, **27**:2025-2026.

Summary

Relation extraction methods for biomedical texts play a crucial role in automatically gathering facts and evidences necessary for life sciences. Although many approaches have been proposed, extracting relations from biomedical text remains a big issue due to, among others, the quality of the extracted relations, computational performance time, and the type of relations being extracted. First, the performance of extraction systems, which is measured in terms of precision, needs to be improved to satisfy the demand of tasks such as building high quality biological databases. Second, most of the proposed systems require a significant computational performance time when applied for large scale extraction. Therefore, these systems are not ready for real time application. Finally, existing approaches have mainly focused on extracting PPIs, and recently on biological events; many relation types are still untouched.

As an attempt to contribute to this research field, in this thesis we have investigated three relation extraction tasks with various settings such as relation types and the availability of training data. By studying methods to extract these relation types, we can understand their common characteristics and requirements. This helps designing extraction methods that can generalize well for the other types. By varying the availability of training data, we can understand how training data influence the performance of extraction techniques. In particular, the main results of this thesis are as follows:

We introduce a novel method to extract and combine relationships between HIV drugs and mutations in viral genomes. Our extraction method is based on natural language processing (NLP) which produces grammatical relations and applies a set of rules to these relations. The results show that our system achieves good performance when compared with the expert systems. Our system is being deployed in the ViroLab project (www.ViroLab.org) to help virologists find evidence for HIV drug resistance from literature in a controlled and automated way.

We propose a hybrid system that employs both rule- and ML-based method to extract protein-protein interactions from texts. By introducing a data partition strategy, we can significantly reduce the number of features used by the ML classifier which then increases the robustness of the system. We demonstrate that our system achieves a performance that is comparable with the state-of-the-art systems when evaluated using 10-fold cross validation. Furthermore, it outperforms these systems when evaluated using cross-corpora criteria, in which, training data and testing data might have completely different properties, meaning that this setting is closer to the real world situation. Furthermore, our system also achieves the best performance in terms of computational efficiency.

We present a novel rule-based method to extract biomedical events from text which consists of two phase: a learning phase and an extraction phase. By using a structured representation, we can decompose the complex and nested structures of biomedical events into simple syntactic layers. Based on this structure, the system both learns rules to extract events from training data and applies rules to extract events from text. This representation not only allows us simplifying the learning and extraction phases but also requires less syntactic analysis of input sentences. The evaluation results show that our system performs well on both abstract and full text datasets. Furthermore, it achieves superior performance in terms of computational efficiency, ranging from 150 to 200-fold faster than the state-of-the-art systems. It is clearly suited for large-scale experiments. In addition, our approach is simple and generic therefore it can easily be adapted to extract any types of relations.

Samenvatting

Methodes voor het extraheren van relaties in biomedische teksten spelen een cruciale rol in het automatisch vergaren van feiten en bevindingen die nodig zijn voor levenswetenschappen. Hoewel vele mogelijke benaderingen zijn gepresenteerd in de huidige literatuur blijft het extraheren van relaties een groot probleem. Dit komt onder andere door de kwaliteit van de relaties, benodigde computerkracht, en de verschillende types van relaties die mogelijk zijn. Ten eerste moet de 'precisie' van extractiesystemen worden bevorderd om te voldoen aan de standaarden voor bijvoorbeeld het bouwen van databases met biologische informatie. Ten tweede kosten de meeste bestaande extractiesystemen teveel computerkracht waardoor ze niet kunnen worden toegepast op de literatuur op grote schaal. Hierdoor zijn de huidige systemen nog niet klaar voor real-time toepassingen. Ten derde richten de meeste bestaande methoden zich specifiek op interacties tussen proteines (PPI); vele andere types van relaties zijn nog niet of nauwelijks bestudeerd.

In mijn dissertatie lever ik een bijdrage aan dit onderzoeksgebied door het bestuderen van drie verschillende relatie-extractietaken onder invloed van o.a. de beschikbaarheid van trainingsdata en de verschillende types van relaties. Door het bestuderen van de methodes om deze types van relaties te extraheren kunnen we meer begrijpen over de eigenschappen die zij gemeen hebben. Dit helpt bij het ontwikkelen van generieke extractiemethodes die toepasbaar zijn op meerdere relatietypes. Door het variëren van de beschikbaarheid van data om de modellen mee te kalibreren kunnen we beter begrijpen hoe dit de prestaties van extractiemethodes beïnvloedt. In het bijzonder zijn mijn meest belangrijke resultaten als volgt.

We introduceren een nieuwe methode voor het extraheren en combineren van relaties tussen HIV-medicijnen en mutaties in de genetica van het virus. Onze extractiemethode is gebaseerd op 'natural language processing' (NLP) dat grammaticale relaties produceert en een lijst van regels toepast op deze relaties. Uit onze resultaten blijkt dat ons systeem goed presteert in vergelijking met andere prominente systemen. Ons systeem wordt toegepast in het ViroLab project (www.ViroLab.org) om virologen te helpen bij het vinden van relaties tussen HIV-medicijnen en de mogelijke mutaties van het HIV-virus in een gecontroleerde en geautomatiseerde manier.

We presenteren een hybride systeem dat gebaseerd is op zowel het toepassen van een vaste lijst van regels als het gebruik van kunstmatige intelligentie ('machine learning') om PPIs te extraheren uit de literatuur. We introduceren een strategie voor het opdelen van de gegevens die de complexiteit van de kunstmatige-intelligentie-algoritme significant vermindert, hetgeen de robuustheid van het systeem bevordert. We demonstreren dat ons systeem vergelijkbaar presteert met de state-of-the-art systemen met tienvoudige kruisvalidatie. Bovendien scoort ons systeem beter onder

het ‘cross corpora’ criterium, waar de trainingsdata en de testdata hele verschillende eigenschappen kunnen hebben, hetgeen meer lijkt op de realiteit. Verder vergt ons systeem de minste computerkracht.

We presenteren een nieuwe op regels gebaseerde methode voor het extraheren van biomedische gebeurtenissen uit tekst die bestaat uit twee fases: een leerfase en een extractiefase. Door het gebruik van een gestructureerde representatie delen we de complexe en geneste structuren van biomedische gebeurtenissen op in eenvoudigere syntactische lagen. Op basis van deze structuur leert het systeem nieuwe regels uit de trainingsdata en past de regels ook toe om gebeurtenissen te extraheren. Deze representatie stelt ons niet alleen in staat om de leerfase en de extractiefase te vereenvoudigen, maar vergt ook minder syntactische analyse van de zinnen in de invoertekst. De resultaten van de evaluatie laten zien dat ons systeem goed presteert zowel op samenvattingen als op data bestaande uit volledige teksten van artikelen. Bovendien behaalt het de beste computationele efficiëntie: het is 150 tot 200 keer sneller dan de huidige state-of-the-art systemen. Verder is onze benadering simpel en generiek en kan daarom gemakkelijk worden aangepast voor het extraheren van elk ander type van relaties.

List of publications

1. Bui QC, Nualláin BO, Boucher CA, Sloot PMA: **Extracting causal relations on HIV drug resistance from literature.** *BMC Bioinformatics* 2010, **11**:101.
2. Bui, QC, Katrenko S, Sloot P M A: **A hybrid approach to extract protein-protein interactions**. *Bioinformatics 2011*, 27(2), 259-65.
3. Bui QC, Sloot PMA: **Extracting biological events from text using simple syntactic patterns**. In *Proceedings of BioNLP Shared Task 2011 Workshop*. 2011:143-146.
4. Bui QC, Sloot PMA: **A robust approach to extract biomedical events from literature**. *Bioinformatics 2012*. DOI: 10.1093/bioinformatics/bts487

Acknowledgements

First and foremost, I would like to express my deep gratitude to my promoter and supervisor, Professor Peter Sloot, for accepting me as a PhD student in your group. Peter, you introduced me to the biomedical text mining field and gave me full of freedom to explore this new research field. I highly appreciate your advice, patience, and kind support to edit my writings. I have learned a lot from you.

I would like to thank Arend, Niels, Peter, Erik, and Petra for their enormous help and enthusiastic support in administrative procedures. Thank Breanndán for your supervision in the Virolab project. I would like to extend my sincere thanks to Rick, Mikoto, and Sophia for your constructive comments on my papers. Thanks to all colleagues at the SCS group for sharing great memories during my PhD research time.

I am grateful to all international and Vietnamese friends in Amsterdam for your sharing and encouragement in social life, unforgettable parties, and organizing many sport activities. You all make me feel home.

Last but not least, I would like to thank my family for your support and understanding. Thanks to my sisters and brother for taking care of mom at home. My biggest thank goes to my wife Ha for her love. Honey, you have changed my life!

Amsterdam, July 2012.