BIOEVE

BIOEVE

Protein-Protein Relation Extraction from Biomedical Abstracts

Syed Toufeeq Ahmed
Vanderbilt University

Hasan Davulcu, Sukru Tikves, Radhika Nair, and Chintan Patel
Arizona State University

A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

Library of Congress Cataloging-in-Publication Data:

BioEve / Syed Toufeeq Ahmed . . . [et al.].
p. cm.—(Wiley series in survey methodology)
"Wiley-Interscience."
Includes bibliographical references and index.
ISBN 0-471-48348-6 (pbk.)
1. Surveys—Methodology. 2. Social sciences—Research—Statistical methods. I. Groves, Robert M. II. Series.

HA31.2.S873 2007
001.4'33—dc22 2004044064
Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

CONTENTS IN BRIEF

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

PROTEIN-PROTEIN RELATION EXTRACTION FROM BIOMEDICAL ABSTRACTS

Syed Toufeeq Ahmed, Ph.D.[1], Hasan Davulcu, Ph.D.[2], Sukru Tikves[2], Radhika Nair[2], and Chintan Patel[2]

[1]Vanderbilt University
[2]Arizona State University

1.1 INTRODUCTION

Human genome sequencing marked beginning of the era of large-scale genomics and proteomics, which in turn led to large amount of information. Lots of that exists (or generated) as unstructured text of published literature. The first step towards extracting event information, in biomedical domain, is to recognize the names of proteins [18, 4], genes, drugs and other molecules. The next step is to recognize relationship between such entities [5, 30, 19] and then to recognize the bio-molecular interaction events with these entities as participants [43, 40]. However, several issues make extracting such interactions and relationships difficult since [38]:

1. The task involves free text - hence there are many ways of stating the same fact

2. The genre of text is not grammatically simple

3. The text includes a lot of technical terminology unfamiliar to existing natural language processing systems

4. Information may need to be combined across several sentences, and

5. There are many sentences from which nothing should be extracted.

Information Extraction (IE) [9, 23, 34, 17] is the extraction of salient facts about pre-specified types of events, entities [8] or relationships from free text. Information extraction from free-text utilizes shallow-parsing techniques [14], Parts-of-Speech tagging [7], noun and verb phrase chunking [27], verb subject and object relationships [14], and learned [9, 13, 38] or hand-build patterns to automate the creation of specialized databases. Manual pattern engineering approaches employ shallow parsing with patterns to extract the interactions. In the [30] system, sentences are first tagged using a dictionary based protein name identifier and then processed by a module which extracts interactions directly from complex and compound sentences using regular expressions based on part of speech tags. IE systems look for entities, relationships among those entities, or other specific facts within text documents. The success of information extraction depends on the performance of the various subtasks involved. The SUISEKI system of Blaschke [4] also uses regular expressions, with probabilities that reflect the experimental accuracy of each pattern to extract interactions into predefined frame structures. GENIES [16] utilizes a grammar based *Natural Language Processing* (NLP) engine for information extraction. Recently, it has been extended as *GeneWays* [33], which also provides a Web interface that allows users to search and submit papers of interest for analysis. The BioRAT system [12] uses manually engineered templates that combine lexical and semantic information to identify protein interactions. The *GeneScene* system [25] extracts interactions using frequent preposition-based templates.

Grammar engineering approaches, on the other hand use manually generated specialized grammar rules [32] that perform a deep parse of the sentences. Temkin [41] addresses the problem of extracting protein interactions by using an extend able but manually built *Context Free Grammar* (CFG) that is designed specifically for parsing biological text. The *PathwayAssist* system uses an NLP system, *MedScan* [29], for the biomedical domain that tags the entities in text and produces a semantic tree. Slot filler type rules are engineered based on the semantic tree representation to extract relationships from text. Recently, extraction systems have also used link grammar [20] to identify interactions between proteins [15]. Their approach relies on various linkage paths between named entities such as gene and protein names. Such manual pattern engineering approaches for information extraction are very hard to scale up to large document collections since they require labor-intensive and skill-dependent pattern engineering. Machine learning approaches have also been used to learn extraction rules from user tagged training data. These approaches represent the rules learnt in various formats such as decision trees [11] or grammar rules [42]. Craven et al [13] explored an automatic rule-learning approach that uses a combination of FOIL [31] and *Naive Bayes* Classifier to learn extraction rules.

The BioNLP'09 shared task [1] involved recognition of bio-molecular events, which appear in the GENIA corpus. We mainly focused on task 1, which was detection of an event and its participants.

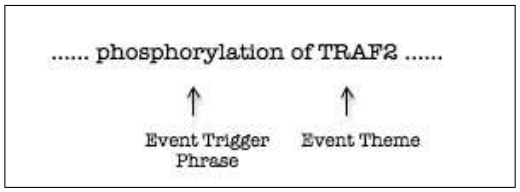

Figure 1.1 Example of Phosphorylation Event

The rest of the chapter is organized as follows. In Section 1.2 we describe BioEve system, Section 1.3 explains in detail different classification approaches, and event extraction using dependency parse tree of the sentence is explained in Section 1.4. Sections 1.5 describes experiments with classification approaches, event extraction and evaluation results for BioNLP'09 shared task 1 [1]. Section 1.6 concludes the paper.

1.2 BIOEVE: BIO-MOLECULAR EVENT EXTRACTOR

A bio-event could be described as a change on the state of a bio-molecule or bio-molecules. An example of an event is shown in figure 1.1. BioEve architecture is shown in figure 1.2. First the biomedical abstracts are split into sentences, before being sent to sentence level classifier. We used *Naive Bayes* classifier to classify sentences into different event class types. Classification at sentence level is a difficult task, as sentences have lesser information as compared to the whole document. To help event extraction module, each of these sentences are then semantically labeled with additional keywords. We created a dictionary-based labeler, which included trigger words from training data, along with the corresponding event type. These labeled sentences are parsed using a dependency parser to identify `argument-predicate` roles. For each event class type, we hand crafted high coverage extraction rules, similar to Fundel *et al.* [19], to identity all event participants. For BioNLP shared task, the event-participant output was formatted to GENIA format.

1.2.1 Bio-Entity Tagging

The first step in extracting bio-events, is to identify candidate participants and the classes to which they belong. The intent is to capture entity type relationships to facilitate queries which is difficult using simple keyword search. An example could be "What are all genes related to eye disorders?" An abstract may contain the term "conjunctivitis" which is a type of eye disorder, but not the actual term "eye disorders". Such results would be missed out if we focus on syntactic term matching approach.

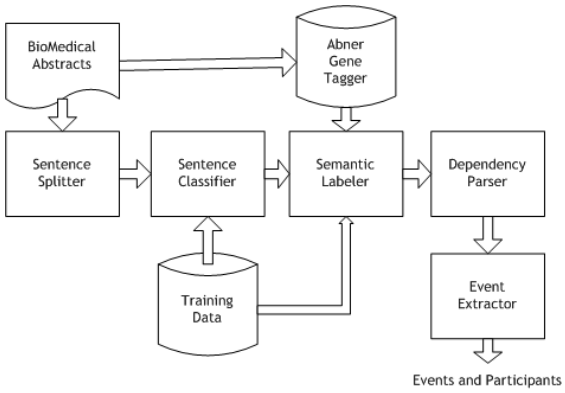

Figure 1.2 BioEve System Architecture

We applied *A Biomedical Named Entity Recognizer* (ABNER) [37], an open source software tool for molecular biology text mining, to tag different gene types including protein names, DNA, RNA, cell line and cell types. Abstract were also found to contain drugs and chemicals which could also participate in an event. We used *Open Source Chemistry Analysis Routines* (OSCAR3)[35] to identify chemical names and chemical structures. Capturing "ISA" relationships gives a single level semantic relationship. To capture an ontology relationship, we used *Unified Medical Language System* (UMLS) [6] MeSH database. A strict matching approach was used to tag valid MeSH terms present in an abstract.

1.2.2 Event Trigger Identification and Classification

A bio-event can be broadly defined as a change on the state of a bio-molecule or bio-molecules, e.g. *phosphorylation of IkB involves a change on the protein IkB*. A relationship generally involves two participants, however a bio event can involve one or more participants, where participants could be entities or events. An event is characterized by a trigger word, which indicates presence of an event, and extracting the bio-medical entities associated with these events. This module is an enhancement of event detection and typing BioNLP'09 Shared Task [1]. We had applied a dictionary-based semantic labeler for this shared task. Further research on this problem highlighted three different approaches of detecting and typing events at various levels of granularity.

1.3 SENTENCE LEVEL CLASSIFICATION AND SEMANTIC LABELING

First step towards bio-event extraction is to identify phrases in biomedical text which indicate presence of an event. The labeled phrases are classified further in to 9 event types. The aim of marking such interesting phrases is to avoid looking at the entire

text to find participants. Full parsing of biomedical literature would be very expensive especially for large volume of text. We intend to mark phrases in biomedical text, which could have a potential event, to serve as a starting point for extraction of event participants. BioEve event extraction module depends on class labels for extraction. To help with this task, we needed to improve sentence labeling with correct class type information. For this, we employed dictionary based semantic class labeling by identifying trigger (or interaction) words, which clearly indicate presence of a particular event. We used ABNER [37] gene name recognizer to enrich the sentences with gene mentions. There have been cases in the training data where the same trigger word is associated with more than one event type. To resolve such cases, the trigger words were mapped to the most likely event type based on their occurrence count in the training data. We labeled trigger words in each sentence with their most likely event type. These tagged words served as a starting point for the extraction of event participants. This was done to speed-up the extraction process, as event extraction module now only needs to focus on the parts of the sentences related to these tagged trigger words.

1.3.1 Incremental Approach towards Classification Task

For the classification problem at hand, we started with most popular and simple algorithm first (*Naive Bayes*) and incrementally moved to more sophisticated machine classification algorithms. Findings and observations at each level were used as a learning for improvements at the next level of experiments. Table 1.1 gives an overview of different classifiers applied at different levels of granularity and the features used by these classifiers. We first started with identification of a single label per sentence, further advancing to multiple labels per sentence and eventually marking phrases in text and classifying these phrases.

1.3.2 Single Label, Sentence-Level Classification

This approach was a preliminary attempt towards understanding problem at hand and identifying features suitable for the classification. We used *Naive Bayes* classifier as a baseline, since it is known to perform well for text classification and is fast and easy to implement. Bayesian classiers assign the most likely class to a given example described by its feature vector. Learning such classiers can be greatly simplified by assuming that features are independent given class, that is,

$$P(X|C) = \sum_{i=1}^{n} P(X_i|C)$$

where $X = (X_1, \cdots, X_n)$ is a feature vector and C is a class.

For training the classifier, every sentence in the abstract was treated as a separate instance.The class label for a sentence was based on the most frequent event type occurring in the sentence. If there is a single dominant event int he sentence the instance is labeled with that event type. If there is more than one event in a training instance, then the first encountered event type is passed to the classifier for that

Table 1.1 Summarization of Classification Approaches

Granularity	Features	Classification Approaches
Single Label, Sentence Level	Bag - of - words (BOW) BOW + gene names boosted BOW + trigger words boosted BOW + gene names and trigger words boosted	*Naive Bayes*
Multiple Labels, Sentence Level	BOW	*Naive Bayes* + *Expectation Maximization* *Maximum Entropy*
Event trigger phrase labeling	BOW + 3gram and 4 gram prefixes and suffixes + orthographic features + trigger phrase dictionary	*Conditional Random Fields* (CRF)

instance. We used WEKA [3], a collection of machine learning algorithms for data mining tasks, for identifying single label per sentence approach. As WEKA does not support multiple labels for same instance, we had to include a trade-off here, by including the first encountered label in the case where the instance had multiple labels.

For the feature sets mentioned below, we used the TF-IDF representation. Each vector was normalized based on vector length. Also, to avoid variations, words / phrases were converted to lowercase. Based on WEKA library token delimiters, features were filtered to include those which had an alphabet as a prefix, using regular expressions. For example, features like $-300bp$ were filtered out, but features like $p55$ which is a protein name, were retained. We experimented with the list of features described below, in order to understand how well each feature suits the corpus under consideration.

- Bag –of-words model: This model classified sentences based on word distribution.

- Bag-of-words with gene names boosted: The idea was to give more importance to words, which clearly demarcate event types. To start with, we included gene names provided in the training data. Next, we used the ABNER gene tagger to tag gene names, apart from the ones already provided to us. We boosted weights for renamed feature “protein” by 2.0.

- Bag-of-words with event trigger words boosted:

We separately tried boosting event trigger words. The list of trigger words was obtained from training data. This list was cleaned to remove stop words. Trigger words were ordered in terms on their frequency of occurrence with respect to an event type, in order to capture trigger words which are most discriminative.

- Bag-of-words with gene names and event trigger words boosted: The final approach was to boost both gene names and trigger words together. Theoretically, this approach was expected to do better than previous two feature sets discussed. Combination of discriminative approach of trigger words and gene name boosting was expected to train the classifier better.

1.3.3 Multiple Labels, Sentence-Level Classification

Based on heuristics, the GENIA corpus data set on an average has more than 1 event per sentence. There were instances in training data which had a single dominant event. However in some cases, multiple event types occurred in a training instance with an equal probability. Hence, there is a need to consider multiple labels per sentence. Instead of strictly classifying a sentence under one label, the intent is to determine event type probability in the sentence. To explain this further, consider the example in Figure 1.3. The phrases italicized indicate trigger phrases, where the phrases $blocked$ and $prevented$ indicate presence of $Negative\ regulation$ event, $proteolytic\ degradation$ identifies $Protein\ catabolism$ event. $Negative\ regulation$ is a dominant event type in this sentence, however, the sentence also talks about other event types like $Protein\ catabolism$, although with a lesser probability. For a user looking for content related to $Protein\ catabolism$, could be interested in sentences like 1.3. Based on analysis of PUBMED abstracts, we considered a threshold of 0.2 probability.

Furthermore, sodium salicylate $blocked$ the LPS-induced $proteolytic\ degradation$ of I kappa B alpha, which $prevented$ the nuclear $translocation$ of c-Rel/p65 heterodimers.

Figure 1.3 Plain Text Sentence

We used classification algorithms from MALLET library [2]. Biomedical abstracts are split into sentences. For training purposes, plain text sentences are transformed into training instances as required by MALLET. The classifier is trained based on these formatted instances. Test abstracts are converted to instances as well and the trained classifier predicts probability of each event type for every sentence. A threshold of 0.2 probability was applied to identify the top event types present in the sentence. Use of the classifiers under MALLET requires data transformation in to formatted training instances. For multiple labels for sentence, experimented with *NaiveBayesEM* (basic *Naive Bayes* classifier, which utilizes *Expectation Maximization* to facilitate the classification) and *Maximum ENTropy* (MaxENT) classifier.

Maximum entropy is a probability distribution estimation technique [28]; where the underlying principle of maximum entropy is that without external knowledge, one should prefer distributions that are uniform [28]. Labeled training data is used to derive a set of constraints for the model that characterize the class-specific expectations for the distribution [28]. The two main aspects of *Maximum Entropy* classifier are feature selection and parameter estimation. Feature selection part selects the most important features of the log-linear model, and the parameter estimation part assigns a proper weight to each of the feature functions [21]. *Maximum Entropy* estimation produces a model with the most uniform distribution among all the distributions satisfying the given constraints [21]. The feature set used was bag-of-words model approach.

1.3.4 Phrase Level Labeling

The next level of improvement was to advance from sentence level labeling to phrase level labeling. This is more accurate since we are not only identifying event types present in a sentence, but also marking their positions in the text. In this approach, we considered event trigger phrase classification as a sequence segmentation problem, where each word is a token in a sequence to be assigned a label [36].

Based on examples from training data, following were some of the key observations made, which proved to be beneficial while training the phrase level classifier:

- Not all events are tagged in GENIA corpus. Set of proteins and certain type of genes were selected and only events related to these selected proteins were tagged. Consider the example in Figure 1.4. In this example, the word *inhibition* is labeled as belonging to *Negative_regulation* event type. In second example shown in Figure 1.5, even though it closely resembles the example in Figure 1.4, it was not labeled in the training data, because *NF-kappa B* was not selected in the list of proteins for abstract ID 8096091.

- Taking context in to consideration was important while marking trigger words. Figure 1.6 gives two examples of *Transcription* and *Phosphorylation* event types respectively, which are valid in given context. Figure 1.7 indicate examples for trigger words *transcription* and *phosphorylation* which are not valid trigger words in the given context.

· · · Cytokine rescue from glucocorticoid induced apoptosis in T cells is mediate through **inhibition** of *IkappaBalpha*. · · ·

Figure 1.4 Selected Events Annotation (PUBMED Abstract ID: 9488049)

· · · p65 restores intracellular inhibition of NF-kappa B · · ·

Figure 1.5 Valid Event Not Labeled (Abstract ID: 8096091)

· · · leading to NF-kappaB nuclear translocation and **transcription** of E-selectin and IL-8 · · ·

· · · Ligation of CD3 also induces the tyrosine **phosphorylation** of HS1 · · ·

Figure 1.6 Valid Event Phrases Considering Context

· · · requires expression of cytokines and chemokines as well as activation of the **transcription** factor nuclear factor (NF)-kappaB

· · ·. Protein **phosphorylation** has an important role in the regulation of these two factors · · ·

Figure 1.7 Invalid Event Phrases Considering Context

1.3.5 Conditional Random Fields Based Classifier

Conditional Random Fields (CRFs) are undirected statistical graphical models, a special case of which is a linear chain that corresponds to a conditionally trained finite-state machine [36]. CRFs in particular have been shown to be useful in parts-of-speech tagging [24], shallow parsing [39], and named entity recognition for news wire data [26]. We customized ABNER [37], which is based on MALLET, to suit our needs. ABNER employs a set a orthographic and semantic features. As an improvement to the approaches discussed so far, we intended to include biomedical domain information while training the classifier. We analyzed the features used by ABNER for protein and gene name recognition.

1.3.6 Feature Selection

We utilized both orthographic, and semantic features in training the system. The orthographic features were extracted from BIONLP-NLPBA 2004 shared task vocabulary, while the semantic features were incorporated through ABNER.

1.3.6.1 Orthographic features The default model included the training vocabulary (provided as part of the BIONLP-NLPBA 2004 shared task) in the form of 17 orthographic features based on regular expressions [36]. These include upper case letters (initial upper case letter, all upper case letters, mix of upper and lower case letters), digits (special expressions for single and double digits, natural numbers and real numbers), hyphen (special expressions for hyphens appearing at the beginning and end of a phrase), other punctuation marks, Roman and Greek words, and 3-gram and 4-gram suffixes and prefixes.

1.3.6.2 Semantic Features ABNER uses semantic features are provided in the form of hand-prepared and database referenced lexicons. Table 1.2 gives information about the basic lexicon groups used. This information is referenced from [36].

Table 1.2 Feature Selection

Lexicon Description	Source	Lexicon Count
Greek letters, amino acids, chemical elements, known viruses, abbreviations of all these	Entered by hand	7
Genes, chromosome locations, proteins, and cell lines	Online public databases	4
Lexicons for CELL_TYPE	Google web index	30

1.3.7 Trigger Phrase Dictionary

Based on the GENIA training data, a trigger phrase dictionary was created, providing the mapping between a trigger phrase and event type(s). This list was cleaned to remove stop words. The stop word cleaning was applied for single word trigger phrases which are included a stop word list. All possible morphological forms of trigger words were added to the list; e.g. for trigger word $upregulation$, terms like $upregulates$ and $upregulated$ were added as well.

The list was first ordered to identify the discriminating trigger phrases for each event type. An event type was associated with a trigger phrase based on the number of times an event type is associated with that trigger word. Finally, filtered trigger words are ordered such that multi-word phrases are tagged in preference to phrases with single word, e.g. phrase $gene\ expression$ indicates presence of $Gene\_expression$ event as compared to single trigger phrase $expression$. The dictionary of trigger words was selectively applied, based on knowledge about false positives from training data.

1.4 EVENT EXTRACTION USING DEPENDENCY PARSING

The sentences, after being class labeled and tagged, are parsed using a dependency parser (Stanford parser [10]) to identify `argument-predicate` roles. Words in the sentence and the relationships between these words form the dependency parse tree of the sentence. One problem encountered during initial testing stages was due to the gene and protein names. These names are not a part of the standard English dictionary and as a result, the dependency parses of the sentences gives unexpected results. To remedy the situation, each mention is substituted by a unique identifier. For example, *PU.1* would be substituted by *T7*, depending on its occurrence in the text. The annotations are not part of the standard English dictionary either, but they do not cause the dependency parser to parse the sentence incorrectly and also, searching for them in the dependency tree can be simplified by simple regular ex-

pressions. For our system, we used typed-dependency representation output format from Stanford parser which is a simple tuple, `reln(gov, dep)`, where `reln` is the dependency relation, `gov` is the governor word and `dep` is the dependent word. Consider the following example sentence:

> We investigated whether PU.1 binds and activates the M-CSF receptor promoter.

After this sentence is class labeled and tagged:

> We investigated whether T7 binds/*BINDING* and activates/*POSITIVE_REGULATION* the T8 promoter.

The tagged sentence is parsed to obtain dependency relations as shown below:

```
nsubj(investigated-2, We-1)
complm(binds-5, whether-3)
nsubj(binds-5, T7-4)
ccomp(investigated-2, binds-5)
conj_and(binds-5, activates-7)
det(promoter-10, the-8)
nn(promoter-10, T8-9)
dobj(binds-5, promoter-10)
```

This sentence mentions two separate events, *binding* and *positive regulation*. Let's consider the extracting the event *binding* and its participants. Figure 1.8 shows the parse tree representation and the part of the tree that needs to be identified for extracting event *binding*.

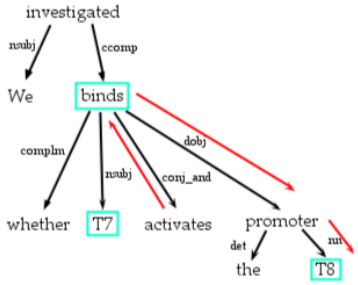

Figure 1.8 Dependency Parse tree, and event "binding" and its participants are shown.

The rule matching begins from the root node of the dependency parse tree. The module searches the tree in a breadth-first fashion, looking for event trigger words. It does not search for occurrences of protein or gene annotations. On finding a trigger word, it marks the node in the tree and activates the rule matcher for the corresponding event class on that node. The matcher searches the tree for participants of the event and on finding them successfully, creates a record in the result set corresponding to the event. For example, in the tree shown above, in figure 1.8, "binds" is a

```
Input: Abstracts tagged with the interaction words and class labels
Output: Bio-molecular events with interaction words and the participants
foreach abstract do Split abstract into sentences
    foreach sentence in current abstract do
        retrieve all the interaction words in current sentence;
        sort them according to precedence of events;
        foreach interaction word in the sentence do
            extract the participants by applying the corresponding event's rule
            to the sentence's dependency parse;
        end
    end
end
```

Figure 1.9 One pass extraction algorithm

trigger word for a binding event. The extraction module fires a signal on detecting its corresponding node in the parse tree. It then marks the node and loads the binding event rule matcher on it. This matcher searches for the participants of the binding event as per the rules created for it. It finds T7 and T8 in the tree and reports them back. This results in the creation of a binding event, with the trigger word "binds" and participants T7 and T8 dereferenced to "PU.1" and "M-CSF receptor".

1.4.1 One-pass Extraction

For each event class type, we carefully hand crafted rules, keeping theme of the event, number of participants, and their interactions into consideration. In an extraction rule, `T` represents the occurrence of protein in sentence. If multiple proteins are involved, then subscripts, $\texttt{T}_n$, are used to represent this. The rule is triggered when it matches `I` (for an *interaction word*, or *trigger word*) in the sentence. Some dependency relations and rule predicates are explained below:

1.4.1.1 Extraction algorithm The algorithm to extract events and participants from the abstracts is shown in figure 1.9. All the abstracts are iterated over once, their text is split into constituent sentences, each sentence is converted to its dependency tree and the rule matcher then works on the dependency tree to extract an event and its participants.

1.4.1.2 Multiple events and Nested events A single sentence may contain multiple event mentions and their respective participants. In case of multiple events per sentence, one of these cases may hold true.

- The sentence mentions multiple and disjoint events
- The sentence mentions multiple and nested (connected) events

Multiple, disjoint events are events which involve separate or the same set of proteins or genes. These events do not encapsulate another event within themselves.

The parse for a sentence with a nested event
advmod(stimulated-7, However-1)
preconj(TNF-4, neither-3)
nsubj(stimulated-7, TNF-4)
conj_or(TNF-4, LPS-6)
nn(expression-9, T9-8)
dobj(stimulated-7, expression-9)
prep_in(stimulated-7, HUAECs-11)

Table 1.3 Dependency parse of a nested event

An example of such an event mention would be the sentence stated as example before: *"We investigated whether T7 binds and activates the T8 promoter"*. In this sentence, "binds" and "activates" are two distinct events, "binds" represents binding and "activates" represents positive-regulation. They are not nested events, because the participants in both are proteins. One event's result is not the participant for another. Even though both act on the same set of proteins, T7 and T8, they are distinct.

Nested events on the other hand, have other events or their products as their participants. These kind of events are difficult to detect. An example of a nested event is: "However, neither TNF or LPS stimulated VCAM-1 expression in HUAECs". The trigger words in the sentence are "stimulated" and "expression". "Stimulated" denotes positive-regulation and "expression" denotes gene-expression. The gene-expression event is catalyzed by the positive-regulation event. This is an example of a nested event.

Extraction of nested events is difficult due to the nature of their parse result. The dependency parse of the sample sentence is given in table 1.3. The event trigger words "stimulated" and "expression" are related to each other. A rule match will be triggered for both these events and both will result in T9, when the rule for "stimulated" should produce the trigger word expression and its corresponding event.

1.4.1.3 Sample parse and extraction This section uses a sample sentence to demonstrate how BioEve extracts events and their participants from plain text. Consider the following sentence:

> *During CD30 signal transduction, we found that binding of TRAF2 to the cytoplasmic domain of CD30 results in the rapid depletion of TRAF2.*

The proteins, "CD30" and "TRAF2" are tagged and their occurrences are replaced with proper annotations. The trigger words are also tagged in the sentence. This results in the following form of the sentence.

> *During T11 signal transduction, we found that binding/BINDING of T12 to the cytoplasmic domain of T13 results in the rapid depletion/NEGATIVE_REGULATION of T14.*

The sentence text that is parsed using the dependency parser is "During T11 signal transduction, we found that binding of T12 to the cytoplasmic domain of T13 results

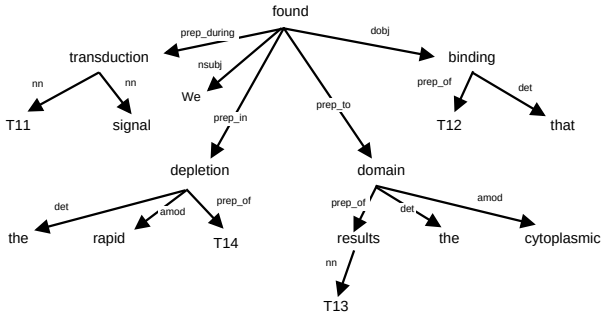

Figure 1.10 Dependency parse tree of the sample sentence

in the rapid depletion of T14". Note that the annotations of the trigger words are removed. This is to prevent the parser from getting confused by the irregular annotation format. Another thing to note is that the event of transduction has not the been tagged even though its corresponding participant has been identified. It's dependency parse tree can be visualized as given in figure 1.10. The extraction procedure will be shown using the tree representation as it is more intuitive.

The extraction module retrieves all the trigger words from the sentence and sorts them as per the even class precedence order. In the sample sentence, "binding" and "depletion" are the trigger words. "Depletion", which suggests negative-regulation has higher precedence than "binding" and hence is searched for first. The extraction module starts from the root of the tree and searches for the event trigger word.

Figure 1.11 shows the rule matcher extracting the instance of a negative regulation event. Starting from the root, the module detects "depletion" at the highlighted node. It knows that this word depicts negative regulation and loads the rules for this event. The first rule for negative regulation is "obj(verb/*T*, *P*)", which means that the trigger word (*T*) is a verb and the protein (P) is its object. The trigger word here, "depletion" is not a verb and hence this rule fails. The module moves to the second rule. This one is "prep(*T*, *P*)", that is, the trigger word and the protein are connected by a preposition. "depletion" and "T14" are connected by a preposition, the word "of". Hence, this rule generates a hit and consequenty the event and its participant are extracted.

After extracting the negative regulation event, the module considers the next event in the order. This sentence has just one left, "binding". It again starts the search from the root and finds the trigger word as highlighted in figure 1.12. The first rule to be matched is P_1 (*T*) P_2, where P_1 and P_2 are the two participant proteins. The rule specifies that the trigger word lies between the nodes for the proteins in the dependency tree. A search for protein annotations on the left tree and right tree of the trigger word node returns a successful match for this rule. The two participant proteins and the trigger word are recorded in the result set as one binding event.

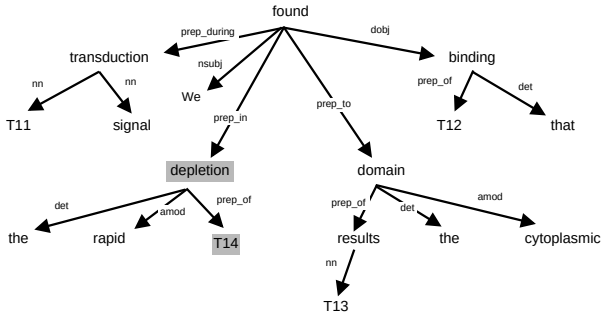

Figure 1.11 Extraction of the negative-regulation event in the given sample sentence. The rule that matches and extracts is "prep(T, P)". T represents the trigger word and P is the protein annotation.

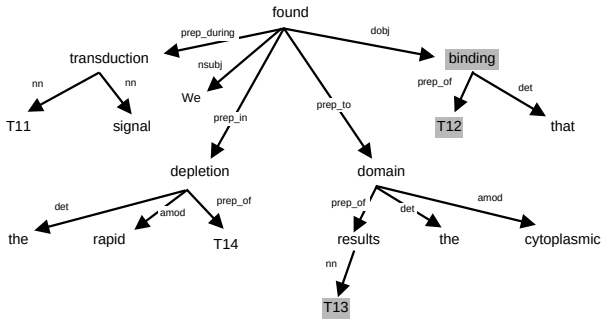

Figure 1.12 Extraction of a binding event in the given sample sentence. The rule that matches and extracts is "P_1 (T) P_2".

```
Input: Abstracts tagged with the interaction words and class labels
Output: Bio-molecular events with interaction words and the participants
foreach abstract do Split abstract into sentences
    foreach sentence in current abstract do
        retrieve all the interaction words in current sentence;
        sort them according to precedence of events;
        First pass, without the regulation events
        foreach interaction word in the sentence do
            extract participants for non-regulation events using the extraction
            rules;
            replace the event trigger words with event annotations;
        end
        Second pass, for the regulation events
        foreach interaction word in the sentence do
            extract the participants, entities or nested events using the rules and
            the replaced annotations;
        end
    end
end
```

Figure 1.13 Two pass extraction algorithm, to handle nested regulation events separately

1.4.2 Two-pass Extraction

Nested events occur as participants for most regulation events. This caused the extraction to give less recall and even lesser precision numbers. To aid this situation, two-pass extraction was used. The precedence order of events is essentially kept the same as one-pass extraction. The difference is that the extraction is done using two passes, the first pass for non-regulation events and the second pass exclusively for regulation events. At the end of the first pass, the events extracted have their trigger words replaced with event annotations so that they may be extracted as themes in the second pass if a rule matches. The algorithm is given in figure 1.13.

1.4.3 Event Extraction Rules

The BioEve system has been designed to extract events and participants from plain text abstracts. Dependency parsing generates the dependency graph on which extraction rules are applied. Due to the fact that the extraction relies solely on plain text, the rules reflect the structure of English grammar. Most of the rules are similar in nature, owing to similarities in the style of writing actions being performed on objects or actions being performed by subjects. The binding event is covered in more details, in a separate section, the three types of regulations in another and the rest of the rules will be clubbed together in a third section. Most of the smaller events have common rules which reflect their grammatical representation. Also, these events involved only a single participating protein and hence the rules are also quite simple.

Rules for a particular event also have precedence order. They are fired from higher to lower order, a reasoning which is based on the rule's accuracy of extracting events.

1.4.4 Binding

Binding events typically involve two proteins. They may involve more that two participants but these cases are rare and hence were ignored for the rule set. The first three rules consider a pair of participants. Trigger words for binding are usually in noun or verb form. In the noun form, the event will be talking about two object clauses. Searching the object clauses can result in a hit for the proteins. Rule 1 looks for such occurrences. For example, "TRADD was the only protein that interacted with wild-type TES2" has the trigger word, "interacted" occurring between two protein occurrences. Another form the noun trigger word can occur is connected with one protein occurrence, with the other protein connected to the first one. Rules 2 and 3 handle this instance of the phrase. As an example, consider the sentence "... binded with TRADD and TES2 ..." or "... binding of TRADD with TES2 ...".

The rules for binding are listed below, in their order of precedence.

1. P_1 (T) P_2
2. prep(T, P_1); prep(P_1, P_2)
3. prep(T, P_1); conj(P_1, P_2)
4. obj(verb/T, P)
5. prep(T, P)
6. *ConnectedRule*
7. *NearestRule*

Rules 4 and 5 are for the instances with only one participant. In such cases, the trigger word is usually a verb with the participant as its direct object or connected with a preposition. The former case is higher in precedence because it is natural to talk about an action occurring over some object in direct speech in English.

1.4.5 Positive-, Negative- and normal gene Regulation

The three types of gene regulations are considered as separate event types for better classification accuracy and easier extraction. In general, regulation of any type are a collection of processes that are used by cells to transform genes into gene products such as proteins. They involve a single participant. Since they are processes, they appear in written text as verbs, with their participants as direct objects or nouns connected to them with a preposition. The *obj* and *prep* rules for positive regulation, negative regulation and regulation reflect this fact. The regulation events are very likely to have other events as their participants. The initial definition of a bio-molecular event applied only to bio-molecular entities. To overcome this defect of

ignoring nested events, the two-pass extraction was used. Two-pass extraction manages to raise the extraction accuracy and volume for the regulation events.

1.4.6 Phosphorylation, Gene Expression, Protein Catabolism, Transcription and Localization

All of the events in this section are also single participant events. They are simple events, meaning that they specify certain processes or transformations taking place on the proteins. Hence, they are verbs connected with their participant with either a preposition or occur with the preposition as their direct object.

1.4.7 *ConnectedRule* and *NearestRule*

The *ConnectedRule* and *NearestRule* are two default rules, fired in case an event class' own rules do not produce any participants. They have been used in the system to increase recall, without sacrificing too much on precision. The rules showed benefits in a few cases where the sentence was too complex, due to which the dependency parse tree caused the rules to fail.

The *ConnectedRule* states that the trigger word and the matching protein should be directly connected with each other, no matter what the dependency relation. The reasoning behind this rule is that if there is a trigger word connected to a protein directly and none of the rules match it, it is either a relation not covered by the rules or a pattern of the event class which is extremely rare. In any case, it is safe to assume that a direct dependency relation implies that the trigger word describes something about its participant.

The *NearestRule* is a catch-all rule. If all the rules, the *ConnectedRule* included fail, the system searches the dependency tree nodes around the trigger word for an occurrence of a participant. The threshold for search is 5 edge hops.

1.5 EXPERIMENTS AND EVALUATIONS

We evaluated BioEve system and major modules on GENIA event corpus made available as part of BioNLP Shared Task [22]. Training set had 800 abstracts (with 7,499 sentences), Development set had 150 abstracts (with 1,450 sentences) and Test set had total 260 abstracts (with 2,447 sentences) in them.

1.5.1 BioEve at BioNLP Shared Task

BioEve shared task evaluation results for Task 1 are shown in Table 1.5.1. Event extraction for classes *gene-expression, protein-catabolism and phosphorylation* performed better comparatively, where as, for classes *transcription, regulation, positive-regulation and negative-regulation*, it was below par. The reason noticed (in training examples) was that, most of the true example sentences of *positive-regulation*

Event Type	Gold (Match)	Ans. (Match)	*Recall*	*Prec.*	*f-Meas.*
Localization	174 (49)	143 (49)	28.16	34.27	30.91
Binding	347 (60)	190 (60)	17.29	31.58	22.35
Gene-expression	722 (323)	803 (323)	**44.74**	**40.22**	**42.36**
Transcription	137 (17)	133 (17)	12.41	12.78	12.59
Protein-catabolism	14 (8)	29 (8)	**57.14**	27.59	37.21
Phosphorylation	135 (72)	107 (72)	**53.33**	**67.29**	**59.50**
EVT-TOTAL	1529 (529)	1405 (529)	**34.60**	**37.65**	**36.06**
Regulation	291 (33)	521 (33)	11.34	6.33	8.13
Positive-regulation	983 (113)	1402 (113)	11.50	8.06	9.48
Negative-regulation	379 (50)	481 (50)	13.19	10.40	11.63
REG-TOTAL	1653 (196)	2404 (196)	11.86	8.15	9.66
ALL-TOTAL	3182 (725)	3809 (725)	22.78	19.03	20.74

Table 1.4 BioNLP Shared Task Evaluation: Task 1 results using approximate span matching.

or negative-regulation class type were mis-classified as either *phosphorylation or gene-expression.* This calls for further improvement of sentence classifier accuracy.

1.5.2 Semantic Classification and Event Phrase Labeling

Employed classifiers were evaluated based on accuracy, precision and recall. Accuracy of a classifier is the percentage of test sample that are correctly classified. Precision indicates the correctness of the system, by measuring number of samples correctly classified in comparison to the total number of classified sentences. Recall indicates the completeness of the system, by calculating the number of results which actually belong to the expected set of results.

Sentence level-single label classification and Sentence-level Multi label classification approaches were evaluated based on how well the classifier labels a given sentence from a test set with one of the nine class labels.

$$Accuracy = \frac{Number\ of\ sentences\ classified\ correctly}{Total\ number\ of\ sentences} \tag{1.1}$$

$$Precision_C = \frac{Number\ of\ sentences\ classified\ correctly\ under\ class\ label\ C}{Number\ of sentences\ classified\ under\ class\ label\ C} \tag{1.2}$$

$$Recall_C = \frac{Number\ of\ sentences\ classified\ correctly\ under\ class\ label\ C}{Number\ of\ sentences\ which\ belong\ to\ class\ label\ C} \tag{1.3}$$

Document level classification using CRF model, was evaluated based on how well the model tags trigger phrases. Evaluating this approach involved measuring the extent to which the model identifies that a phrase is a trigger phrase and how well it classifies a tagged trigger phrase under one of the nine predefined event types.

$$Precision = \frac{Number\ of\ relevant\ and\ retrived\ trigger\ \ phrases}{Number\ of retrived\ trigger\ \ phrases} \tag{1.4}$$

$$Recall = \frac{Number\ of\ relevant\ and\ retrived\ trigger\ \ phrases}{Number\ of\ relevant\ trigger\ \ phrases} \tag{1.5}$$

Retrieved trigger phrases refer to the ones which are identified and classified by the CRF sequence tagger. Relevant trigger phrases are the ones which are expected to be tagged by the model. Retrieved and relevant trigger words refer to the tags which are expected to be classified and which are actually classified by the CRF model. All the classifiers are trained using GENIA training data and tested against GENIA development abstracts.

The average precision and recall for all the approaches is calculated using a weighted average approach. The reason being test instances are not uniformly distributed. Some of the event types like $Positive\_regulation$ have more test instances as compared to event types like $Protein\_catabolism$. So a weighted approach gives a more accurate picture than simple arithmetic average. Weighted average is calculated based on the following equations:

$$Weighted\_Average\_Precision = \frac{\sum_{i=0}^{9} T_i * P_i}{\sum_{i=0}^{9} T_i} \tag{1.6}$$

$$Weighted\_Average\_Recall = \frac{\sum_{i=0}^{9} T_i * R_i}{\sum_{i=0}^{9} T_i} \tag{1.7}$$

where
T_i = Total number of relevant event phrases for event type i
P_i = Precision of event type i
R_i = Recall of event type i

1.5.2.1 Test Data Distribution Table 1.5 gives the total number of test instances for each event type. These counts are used while calculating weighted average for each approach.

Table 1.5 Event Type Test Data Distribution

Event Type	**Total of Test Instances**
Protein_catabolism	17
Gene_expression	200
Localization	39
Phosphorylation	38
Transcription	60
Binding	153
Regulation	90
Positive_regulation	220
Negative_regulation	125
Total	**942**

Table 1.6 Single Label, Sentence Level Results

Classifier	*Feature Set*	*Precision*
NBC	Bag - of - words	62.39%
	Bag - of - words + Gene name boosting	50.00%
	Bag - of - words + Trigger word boosting	49.92%
	Bag - of - words + Trigger word boosting + Gene name boosting	49.77%
	Bag - of - POS tagged words	43.30%

1.5.2.2 Evaluation of Single-Label Sentence Level Classification This approach assigns a single label to each sentence. For evaluation purposes, the classifier is tested against GENIA development data. For every sentence, evaluator process checks if the event type predicted is the most likely event in that sentence. In case a sentence has more than 1 event with equal occurrence frequency, classifier predicted label is compared with all these candidate event types. The intent of this approach was to just understand the features suitable for this corpus. Classifier evaluated were *NaiveBayesMultinomial* classifier from WEKA library, which is a collection of machine learning algorithms for data mining tasks. WEKA contains tools for data preprocessing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.

1.5.2.3 Evaluation of Multi-Label Sentence Level Classification For Maximum Entropy experiments, we used *MaxENTTrainer* class from MALLET library. Table 1.7 gives the precision-recall statistics for this classifier.

Table 1.7 Multi-label, Sentence Level Results (Maximum Entropy Classifier)

Event Type	***Precision***	***Recall***	***F-Measure***
Phosphorylation	0.97	0.73	0.65
Protein_catabolism	0.81	0.68	0.83
Gene_expression	0.88	0.58	0.74
Localization	0.61	0.69	0.70
Transcription	0.49	0.8	0.61
Binding	0.65	0.62	0.63
Regulation	0.52	0.67	0.59
Positive_regulation	0.75	0.25	0.38
Negative_regulation	0.54	0.38	0.45
Weighted Average	**0.68**	**0.53**	**0.57**

The multi-label classification shows some improvement over single-label classification. Also, MALLET is dedicated to text classification whereas WEKA has more generalized machine learning algorithms covering other media like images. *Maximum Entropy* classifier supersedes *NaiveBayesEM* classifier in every event type. One of the main reasons could be because *Maximum Entropy*, unlike *Naive Bayes* does not assume conditional independence among features. Related work [28] shows that even with words as features and word counts as feature weights, *Maximum Entropy* was found to perform better than *Naive Bayes*.

1.5.2.4 Evaluation of Phrase Level Labeling Evaluation of this approach was focused more on the overlap of phrases between the GENIA annotated development and CRF tagged labels. The reason being for each abstract in the GENIA corpus, there is generally a set of biomedical entities present in it. For the shared task, only a subset of these entities were considered in the annotations, and accordingly only events concerning these annotated entities were extracted. However, based on the observation of the corpus, there was a probable chance of other events involving entities not selected for the annotations. So, we focused on the coverage, where both the GENIA annotations and CRF annotations agree upon. CRF performance was evaluated on two fronts in terms of this overlap.

- Exact boundary matching: This involves exact label matching and exact trigger phrase match.
- Soft boundary matching: This involves exact label matching and partial trigger phrase match, allowing 1-word window on either side of the actual trigger phrase.

A detailed analysis of the results showed that around 3% tags were labelled incorrectly in terms of the event type. There were some cases where it was not cer-

tain whether an event should be marked as $Regulation$ or $Positive\_regulation$. Some examples include *the expression of LAL-mRNA*, where $LAL - mRNA$ is a gene, specifically a DNA type. As per examples seen in the training data, the template of the form $expression\ of < genename >$ generally indicates presence of a $Gene\_expression$ event. Hence, more analysis may be need to exactly filter out such annotations as true negatives or deliberately induced false positives.

1.5.2.5 Comparative Analysis of Classification Approaches Table 1.8 gives comparative view of all approaches. CRF has a good trade-off as compared to *Maximum Entropy* classifier results. As compared to multiple labels, sentence level classifiers, it performs better in terms of having a considerably good accuracy for most of the event types with a good recall. It not only predicts the event types present in the sentence, but also localizes the trigger phrases. There are some entries where ME seems to perform better than CRF; for example in case of *Positive_reguation*, where the precision is as high as 0.75%. However, in this case the recall is very low (just *25%*). The *F-Measure* for CRF indicates that as compared to the other approaches, CRF predicts 80% of the relevant tags, and among these predicted tags, 65% of them are correct.

Table 1.8 Summary of Classification Approaches

Event Type	**NB + EM**			***Maximum Entropy***			**CRF**		
	Precision	*Recall*	*F-Measure*	*Precision*	*Recall*	*F-Measure*	*Precision*	*Recall*	*F-Measure*
Phosphorylation	0.62	0.42	0.53	0.97	0.73	0.65	0.8	0.83	0.81
Protein_catabolism	0.6	0.47	0.53	0.97	0.73	0.83	0.85	0.86	0.85
Gene_expression	0.6	0.41	0.48	0.88	0.58	0.74	0.75	0.81	0.78
Localization	0.39	0.47	0.43	0.61	0.69	0.70	0.67	0.79	0.72
Transcription	0.24	0.52	0.33	0.49	0.8	0.61	0.57	0.78	0.66
Binding	0.56	0.63	0.59	0.65	0.62	0.63	0.65	0.81	0.72
Regulation	0.47	0.69	0.55	0.52	0.67	0.59	0.62	0.73	0.67
Positive_regulation	0.70	0.27	0.39	0.75	0.25	0.38	0.55	0.74	0.63
Negative_regulation	0.42	0.46	0.45	0.54	0.38	0.45	0.68	0.82	0.74
Weighted Average	**0.55**	**0.46**	**0.47**	**0.68**	**0.53**	**0.57**	**0.65**	**0.79**	**0.71**

1.5.3 Event Extraction Module

The results of the extraction of events from texts selected from the GENIA corpus are shown in table 1.9. The evaluation measures used are explained below:

$$Precision = \frac{|\text{Correct events} \cap \text{Extracted events}|}{|\text{Extracted events}|}$$

$$Recall = \frac{|\text{Correct events} \cap \text{Extracted events}|}{|\text{Correct events}|}$$

$$f - measure = \frac{2 \times (Precision \times Recall)}{(Precision + Recall)}$$

To evaluate extraction module only, we ran it on the Training data, which has all the entities annotated. Table 1.9 shows one pass extraction results. Event extraction for classes gene expression, protein-catabolism and phosphorylation performed better comparatively, where as, for transcription, regulation, positive-regulation and negative-regulation, it was below par. The reason noticed (in training examples) was that, most of the true example sentences of positive-regulation or negative-regulation class type were mis-classified as either phosphorylation or gene-expression. Improvement in the classification of the semantic labels might help improve the extraction results. On the extraction side, the rules used by the system were simple considering the to take language versatility. Nested events were responsible for the relatively poor numbers for the regulation events. Table 1.10 shows the results for two pass extraction. Significant improvement was obtained due to two-pass extraction. The numbers for non-regulation events remained relatively constant, whereas the regulation events showed a large improvement.

1.6 CONCLUSIONS

In this chapter, we presented a fully automated system to extract bio-molecular events from bio-medical abstracts. By semantically classifying each sentence to the class type of the event, and then using high coverage rules, BioEve extracts the participants of that event. We showed significantly improved *F-Measure* of our classification and labeling module by 27%, by using *Conditional Random Fields* based classifier instead of *Naive Bayes* classifier. And we have also improved *F-Measure* of event participant extraction module by 14.28%. This experimentation shows that there is great scope for further improvements in all aspects of bio-molecular event extraction.

Event class	*Recall*	*Precision*	*F-Measure*
Localization	61.22	84.29	70.93
Binding	46.14	65.80	54.24
Gene expression	62.20	86.97	72.53
Transcription	62.67	84.35	71.91
Protein catabolism	69.09	85.39	76.38
Phosphorylation	72.73	88.89	80.00
Non-regulation Total	59.08	81.58	68.53
Regulation	14.58	21.37	17.34
Positive regulation	19.56	29.26	23.45
Negative regulation	14.88	22.80	18.01
Total	**35.58**	**51.40**	**42.05**

Table 1.9 BioEve Extraction Module Evaluation - One pass extraction

Event class	*Recall*	*Precision*	*F-Measure*
Localization	69.96	85.98	77.15
Binding	50.00	67.59	57.48
Gene expression	65.25	87.50	74.75
Transcription	67.53	85.31	75.39
Protein catabolism	76.36	86.80	81.16
Phosphorylation	73.33	88.97	80.40
Non-regulation Total	63.02	82.53	71.47
Regulation	36.15	50.81	42.24
Positive regulation	38.41	55.12	45.27
Negative regulation	36.63	53.21	43.39
Total	**48.62**	**66.93**	**56.33**

Table 1.10 BioEve Extraction Module Evaluation - Two pass extraction

1.7 ACKNOWLEDGMENTS

We would like convey our thanks to Sheela P. Kanwar and our colleagues for their help with this research.

REFERENCES

1. Bionlp'09. online - "http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/SharedTask/"
2. Mallet. online - "http://mallet.cs.umass.edu/index.php".
3. Weka. online - "http://www.cs.waikato.ac.nz/ml/weka/".

4. C. Blaschke, MA. Andrade, C. Ouzounis, and A. Valencia. Automatic extraction of biological information from scientific text: protein-protein interaction. In *Proceedings of the AAAI conference on Intelligent Systems in Molecular Biology*, pages 60–7. AAAI, 1999.

5. Christian Blaschke and Alfonso Valencia. The frame-based module of the suiseki information extraction system. *IEEE Intelligent Systems*, 17(2):14–20, 2002.

6. Olivier Bodenreider. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32(Database-Issue):267–270, 2004.

7. Eric Brill. A simple rule-based part-of-speech tagger. In *Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing*, pages 152–155, Trento, IT, 1992.

8. Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial Intelligence in Medicine*, 2003.

9. M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. In *Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, pages 6–11, Menlo Park, CA, 1998. AAAI Press.

10. Marie catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *In LREC 2006*, 2006.

11. J.H. Chiang, H.C. Yu, and H.J. Hsu. GIS: a biomedical text-mining system for gene information discovery, 2004.

12. David P. A. Corney, Bernard F. Buxton, William B. Langdon, and David T. Jones. BioRAT: extracting biological information from full-length papers. *Bioinformatics*, 20(17):3206–3213, 2004.

13. Mark Craven and Johan Kumlien. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 77–86. AAAI Press, 1999.

14. W. Daelemans, S. Buchholz, and J. Veenstra. Memory-based shallow parsing. In *proceedings of CoNLL*, volume 99, pages 53–60. Bergen: Association for Computational Linguistics, 1999.

15. Jing Ding, Daniel Berleant, Jun Xu, and Andy W. Fulmer. Extracting biochemical interactions from medline using a link grammar parser. In *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'03)*, page 467. IEEE Computer Society, 2003.

16. C. Friedman, P. Kra, H. Yu, M. Krauthammer, and A. Rzhetsky. Genies: a natural-language processing system for the extraction of molecular pathways from journal articles. In *Proceedings of the International Conference on Intelligent Systems for Molecular Biology*, pages 574–82, 2001.

17. Marc Friedman and Daniel S. Weld. Efficiently executing information-gathering plans. In *15th International Joint Conference on Artificial Intelligence*, pages 785–791, Nagoya, Japan, 1997.

18. K. Fukuda, A. Tamura, T. Tsunoda, and T. Takagi. Toward information extraction: identifying protein names from biological papers. In *Pac Symp Biocomput*, volume 707, page 18, 1998.

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

20. Dennis Grinberg, John Lafferty, and Daniel Sleator. A robust parsing algorithm for LINK grammars. Technical Report CMU-CS-TR-95-125, Pittsburgh, PA, 1995.

21. Y. Gu, A. McCallum, and D. Towsley. Detecting anomalies in network traffic using maximum entropy estimation. 2005.

22. Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. Overview of bionlp'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, Boulder, Colorado, June 2009. Association for Computational Linguistics.

23. Nickolas Kushmerick, Daniel S. Weld, and Robert B. Doorenbos. Wrapper induction for information extraction. In *Intl. Joint Conference on Artificial Intelligence (IJCAI)*, pages 729–737, 1997.

24. John Lafferty, Andrew McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

25. Gondy Leroy, Hsinchun Chen, and Jesse D. Martinez. A shallow parser based on closed-class words to capture relations in biomedical text. *J. of Biomedical Informatics*, 36(3):145–158, 2003.

26. A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proc. of the 7th Conf. on Natural Language Learning*, pages ?–?, 2003.

27. A. Mikheev and S. Finch. A workbench for finding structure in texts. *Proceedings of the Applied Natural Language Processing (ANLP-97), Washington DC*, 1997.

28. K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *IJCAI-99 workshop on machine learning for information filtering*, pages 61–67, 1999.

29. Svetlana Novichkova, Sergei Egorov, and Nikolai Daraselia. MedScan, a natural language processing engine for MEDLINE abstracts. *Bioinformatics*, 19(13):1699–1706, 2003.

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

31. J. R. Quinlan. Learning logical definitions from relations. *Mach. Learn.*, 5(3):239–266, 1990.

32. Fabio Rinaldi, Gerold Schneider, Kaarel Kaljurand, James Dowdall, Christos Andronis, Andreas Persidis, and Ourania Konstanti. Mining relations in the genia corpus. In *Proceedings of the Second European Workshop on Data Mining and Text Mining for Bioinformatics*, pages 61 – 68, 2004.

33. A. Rzhetsky, I. Iossifov, T. Koike, M. Krauthammer, P. Kra, M. Morris, H. Yu, P.A. Duboué, W. Weng, W.J. Wilbur, et al. GeneWays: a system for extracting, analyzing, visualizing, and integrating molecular pathway data. *Journal of Biomedical Informatics*, 37(1):43–53, 2004.

34. L. Schubert. Can we derive general world knowledge from texts? In *Proceedings of the second international conference on Human Language Technology Research*, pages 94–97. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 2002.

35. B. Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications (NLPBA)*, pages 104–107, 2004.

36. B. Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications (NLPBA)*, pages 104–107, 2004.

37. B. Settles. ABNER: an open source tool for automatically tagging genes, proteins and other entity names in text, 2005.

38. Kristie Seymore, Andrew McCallum, and Roni Rosenfeld. Learning hidden Markov model structure for information extraction. In *AAAI 99 Workshop on Machine Learning for Information Extraction*, 1999.

39. Fei Sha and Fernando C. N. Pereira. Shallow parsing with conditional random fields. In *HLT-NAACL*, 2003.

40. Y. Tateisi, T. Ohta, and J. Tsujii. Annotation of predicate-argument structure of molecular biology text. In *JCNLP-04 workshop on Beyond Shallow Analyses*, 2004.

41. Joshua M. Temkin and Mark R. Gilder. Extraction of protein interaction information from unstructured text using a context-free grammar. *Bioinformatics*, 19(16):2046–2053, 2003.

42. Kyu-Young Whang, Jongwoo Jeon, Kyuseok Shim, and Jaideep Srivastava. Advances in knowledge discovery and data mining, 7th pacific-asia conference, pakdd 2003, seoul, korea, april 30 - may 2, 2003, proceedings. In *PAKDD*, volume 2637 of *Lecture Notes in Computer Science*, pages 148 – 158. Springer, 2003.

43. Akane Yakushiji, Yuka Tateisi, Yusuke Miyao, and Jun ichi Tsujii. Event extraction from biomedical papers using a full parser. In *Pac. Symp. Biocomput*, pages 408–419, 2001.