

# Kernel-Based Learning for Biomedical Relation Extraction

Jiexun Li

*College of Information Science and Technology, Drexel University, Philadelphia, PA 19104.  
E-mail: jiexun.li@ischool.drexel.edu*

Zhu Zhang, Xin Li, and Hsinchun Chen

*Department of Management Information Systems, University of Arizona, Tucson AZ, 85721.  
E-mail: {zhuzhang, xinli, hchen}@eller.arizona.edu*

**Relation extraction is the process of scanning text for relationships between named entities. Recently, significant studies have focused on automatically extracting relations from biomedical corpora. Most existing biomedical relation extractors require manual creation of biomedical lexicons or parsing templates based on domain knowledge. In this study, we propose to use kernel-based learning methods to automatically extract biomedical relations from literature text. We develop a framework of kernel-based learning for biomedical relation extraction. In particular, we modified the standard tree kernel function by incorporating a trace kernel to capture richer contextual information. In our experiments on a biomedical corpus, we compare different kernel functions for biomedical relation detection and classification. The experimental results show that a tree kernel outperforms word and sequence kernels for relation detection, our trace-tree kernel outperforms the standard tree kernel, and a composite kernel outperforms individual kernels for relation extraction.**

## Introduction

Information extraction is an important task in natural language processing (NLP). It is aimed at scanning text for information of interest, including entities and the relations among them. Information extraction has many practical applications, primarily in economic and military domains. Reliable extraction of relations between named entities is still a difficult and unsolved NLP problem. Moreover, the emergence of new application domains often brings new challenges to relation extraction.

In recent years, automatically extracting biomedical information has been the subject of significant research efforts due to the rapid growth in biomedical development and discovery (Shatkay & Feldman, 2003). PubMed, the online portal of the U.S. National Library of Medicine (NLM), is a valuable source of biomedical research findings, including over 16 million articles from Medline and other life-science journals. The size of PubMed abstracts grew at an average rate of 1,760 per day in 2005. Such a huge literature body makes manual inspection of biomedical findings very difficult and time-consuming. Various named entity taggers enable the efficient identification of biomedical entities such as genes and proteins (Bunescu et al., 2005; Hirschman, Yeh, Blaschke, & Valencia, 2005; Settles, 2005). A more challenging task is to identify interentity relations of these biomedical entities from text.

Substantial studies have been developing techniques for extracting biomedical relations such as gene-disease relations, protein interactions, or subcellular localizations. However, most biomedical relation extractors still require manual development of lexicons and parsing rules based on domain knowledge. Recently, statistical learning methods have been introduced to information extraction and have shown promising performance for general corpora. Due to the unique patterns of biomedical relations, techniques designed for extracting relations from general text may not be suitable for the biomedical domain. This study is aimed at designing and examining kernel-based learning methods to extract biomedical relations from literature text.

The remainder of this article is organized as follows. We begin by reviewing existing relation-extraction approaches, and in the following section we raise our research questions. Next, we develop a framework of kernel-based learning for biomedical relation extraction, and introduce different kernel functions. Then we present our experiment on a biomedical corpus. We conclude the study by summarizing key insights and future directions.

---

Received February 22, 2007; revised October 23, 2007; accepted October 26, 2007

© 2008 ASIS&T • Published online 7 February 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/asi.20791

## Literature Review

We categorize biomedical relation extraction approaches into three types: co-occurrence analysis, rule-based approaches, and statistical learning. Under each type, methods vary in how they utilize the lexical, syntactic, and semantic information in texts.

### *Co-occurrence Analysis*

Co-occurrence analysis identifies relations between biomedical entities based on their probabilities of occurrence in articles (Jenssen, Laegreid, Komorowski, & Hovig, 2001; Stapley & Benoit, 2000). These approaches are based on the assumption that if two entities are both mentioned in the same article there is an underlying biological relationship. In most cases, only lexical information (i.e., words) is needed for co-occurrence analysis. Due to their simplicity and flexibility, these approaches have been widely used for relation extraction and can achieve high recall. However, since it can capture little syntactic or semantic information, co-occurrence analysis cannot distinguish relation types, and often achieves low precision.

### *Rule-Based Approaches*

For this approach, researchers have manually developed rules based on syntactic or semantic information to parse relations from text. Syntactic information such as part-of-speech (POS) and syntax structures can be represented via data structures such as parse trees. Syntax parsing approaches extensively utilize syntactic information and rely on syntactic rules for relation extraction (Leroy, Chen, & Martinez, 2003; Park, Kim, & Kim, 2001; Thomas, Milward, Ouzounis, Pulman, & Carroll, 2000; Yakushiji, Tateisi, Miyao, & Tsujii, 2001). These approaches are general, and flexible in terms of the applicable domain. However, existing syntax parsers are rarely able to cover the whole variety of syntactic patterns for relations. Some syntax parsers with large coverage may overgenerate irrelevant parses and lead to incorrect relations. Therefore, parsers that primarily rely on syntactic rules generally achieve poor precision for relation extraction.

Another type of rule-based approach relies more on semantic information in sentences (Friedman, Kra, Yu, Krauthammer, & Rzhetsky, 2001; Pustejovsky, Castaño, Zhang, Kotecki, & Cochran, 2002; Rindfleisch, Tanabe, Weinstein, & Hunter, 2000). Semantic indicators of biomedical relations consist of certain slots of trigger words (e.g., “interact with,” “inhibit,” or “bind to”) manually developed by experts based on their domain knowledge. A semantic parser often represents these trigger words as various relation templates. A pair of biomedical entities whose contextual information satisfies a certain predefined semantic template is identified as a relation. Many semantic parsers have reported higher precision than syntax parsers. However, semantic parsers are also subject to poor coverage of templates. Moreover, semantic templates are largely based on domain-specific lexicons, and therefore have lower portability across different domains.

To address the limitations of these two approaches, hybrid parsers have been developed to take advantage of both syntactic and semantic information. In most hybrid approaches (Gaizauskas, Demetriou, Artymiuk, & Willett, 2003; Novichkova, Egorov, & Daraselia, 2003), syntactic analysis takes place first to create possible parses from the original sentence. Next, they eliminate incorrect parses and identify domain words (such as genes) based on semantic information. Unfortunately, semantic analysis after syntactic processing still cannot effectively improve the poor coverage of syntax grammars. McDonald, Chen, Su, and Byron (2004) combined access to syntactic and semantic information via a single grammar and reported higher precision and recall. Such hybrid parsers maintain both the flexibility of syntax parsing and high precision of semantic analysis. However, rule-based approaches require manual encoding of syntactic and semantic rules based on domain knowledge, which is very labor intensive and time-consuming.

### *Statistical Learning*

Relation extraction can be formulated as a text-classification problem. Each entity pair from a sentence is regarded as a candidate relation instance. Sentences that contain multiple entities can derive multiple relation instances. Each relation instance can be represented in a certain format to capture the contextual information of the two entities in the sentence. Relation extraction is aimed at building a classification model to determine whether two entities from a sentence have a relation to each other and, if they do, what type of relation it is. Such a model can be trained by statistical-learning approaches. Unlike rule-based approaches, statistical learning requires little or no manual development of rules or templates. Instead, patterns are automatically learned from a corpus of documents in which human experts have tagged the desired relations. Thus, a model consisting of these patterns can be used to extract relations from new documents. Although the annotation of relations in the corpus still requires manual inspection, the whole framework is highly portable. Statistical learning can be categorized into feature-based methods and kernel-based methods.

*Feature-based methods.* For feature-based methods, each data instance is represented as a feature vector  $X = \{x_1, x_2, \dots, x_n\}$  in an  $n$ -dimensional space. Features are defined and selected to capture the data characteristics. For instance, “bag-of-words” methods represent a piece of text as a vector in which each element indicates the occurrence of a specific word (Donaldson et al., 2003; Mitsumori, Murata, Fukuda, Doi, & Doi, 2006). Rosario and Hearst (2005) compared generative graphical and discriminative models for relation extraction using both word and role features. Bunescu et al. (2005) developed three generalization methods that utilize word or POS sequences as features, and induce rules of protein relations. These methods require features to be explicitly defined and enumerated for a vector representation. Unfortunately, natural language processing (NLP) tasks often involve large

numbers of words, which can lead to high dimensionality but sparse feature vectors. If a sentence is represented as a complex structure such as a parse tree, features cannot be easily defined to capture the structural information. Moreover, features defined on heterogeneous data representations (e.g., bag-of-words and parse trees) capture different information but may be incompatible to each other.

**Kernel-based methods.** Kernel-based methods are an effective alternative to explicit feature extraction (Cristianini & Shawe-Taylor, 2000). They retain the original representation of objects and use the object only via computing a kernel function between a pair of objects. Formally, a kernel function is a mapping  $K: \mathbf{X} \times \mathbf{X} \rightarrow [0, \infty)$  from input space  $\mathbf{X}$  to a similarity score  $K(x, y) = \phi(x) \cdot \phi(y) = \sum_i \phi_i(x)\phi_i(y)$ , where  $\phi_i(x)$  is a function that maps  $\mathbf{X}$  to a higher dimensional space with no need to know its explicit representation. A kernel function is required to be symmetric and positive-semidefinite. Such a kernel function makes it possible to compute the similarity between objects without enumerating all the features. Given a kernel matrix of pair-wise similarity values, a kernel machine, such as a support vector machine (SVM) (Cristianini & Shawe-Taylor), can train a model for future prediction. Kernel-based methods have been frequently used in machine-learning areas such as pattern recognition (Chen, Yuen, Huang, & Dai, 2005; Zhao, Yuen, & Kwok, 2006), data mining (Zhou & Wang, 2005), text mining (Sun, Lim, Ng, & Srivastava, 2004), and Web mining (Yu, Han, & Chang, 2004). The performance of kernel methods are mainly determined by the selection and design of the kernel functions.

In NLP, various kernels have been applied to information extraction. For simple data representations (e.g., bag-of-words), in which features can be easily extracted, some basic kernel functions such as a linear kernel, a polynomial kernel, and a Gaussian kernel are often used. For data in structured representation, convolution kernels are frequently used (Collins & Duffy, 2001; Haussler, 1999). Convolution kernels are a family of kernel functions, including string kernels (Lodhi, Saunders, Shawe-Taylor, Cristianini, & Watkins, 2002), tree kernels (Zelenko, Aone, & Richardella, 2003), and so on. They define the similarity between objects as the convolution of “subkernels,” i.e., the kernels for the decomposition of the objects. String kernels capture the sequence patterns in data instances. Lodhi et al. (2002) defined string kernels on a letter or word sequence in sentences for text classification. Tree kernels capture the structure of a syntactic parse tree and have been applied in relation extraction (Zelenko et al., 2003). Some recent studies have revised these tree kernels by incorporating richer semantic information (Bunescu & Mooney, 2005; Culotta & Sorensen, 2004). In most tree kernels, each relation instance is represented by the minimum subtree that covers the entity pair, which can often capture major contextual information but may lose some useful information in the rest of the tree.

Another advantage of kernel methods is that they transform different data representations into kernel matrices of the

same format, which enables the integration of heterogeneous information (Cristianini & Shawe-Taylor, 2000; Joachims, Cristianini, & Shawe-Taylor, 2001; Lanckriet, De Bie, Cristianini, Jordan, & Noble, 2004). Studies have shown that composite kernels can reduce kernel sparsity and improve learning performance for NLP (Culotta & Sorensen, 2004; Zhao & Grishman, 2005).

### Summary and Research Gaps

Among various biomedical relation extractors, co-occurrence analysis utilizes little contextual information, whereas rule-based approaches based on syntactic and/or semantic information have shown good performance, but require significant manual efforts. Statistical learning can automatically learn relation patterns from annotated corpora. In particular, kernel-based learning methods have shown promise in identifying various social relations such as action-role, part-of, or locational relations between named entities such as people, organizations, and locations from newspaper articles (Bunescu & Mooney, 2005; Zelenko et al., 2003). A major challenge in biomedical relation extraction is that current POS taggers and parsers that were usually trained on general text do not perform well on the biomedical literature (Bunescu, Mooney, Weiss, Schölkopf, & Platt, 2006). Moreover, kernel functions designed for general relation extraction may not be suitable for the biomedical domain. To the best of our knowledge, there have been few studies that systematically evaluated kernel methods for biomedical relation extraction. Therefore, designing a framework of kernel-based methods for biomedical relation extraction is a necessity and a challenge.

### Research Questions

To address these issues, we examine the performances of kernel-based methods for biomedical relation extraction. In this study we only deal with intrasentence relations. Specifically, given a sentence  $s$  consisting of entities  $\{e_1, \dots, e_n\}$ , we aim to identify and classify all relations  $r_{ij} = \langle e_i, e_j \rangle$ . Our study focuses on four research questions:

1. How can we use kernel-based learning to extract biomedical relations from literature text?
2. Which data representation of instances is better for kernel-based relation extraction?
3. How can we modify kernels to improve the performance of relation extraction?
4. Can the composition of kernels improve the performance of relation extraction?

### Kernel-Based Learning for Biomedical Relation Extraction

We develop a framework of kernel-based learning for biomedical relation extraction as shown in Figure 1. This framework can be decomposed into four major modules: entity recognition, relation annotation, kernel construction, and learning and evaluation.

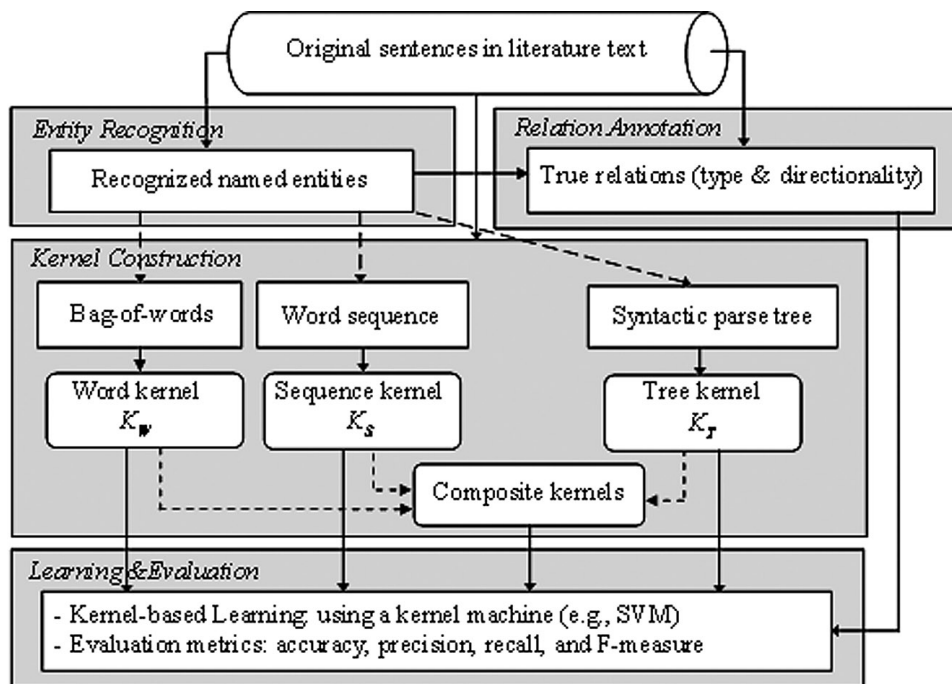


FIG. 1. A framework of kernel-based learning for relation extraction.

### Entity Recognition

In order to extract relations between two entities, a prerequisite step is the identification of named entities from text. Specifically, biomedical relation extraction first requires recognizing biomedical entities such as genes, proteins, and diseases in sentences. The entity recognition process can be performed in either manual or automatic fashion. The manual approach requires domain experts to identify named entities based on their knowledge, and annotate them in sentences. Such a manual process can assure a high degree of correctness, but it can be time-consuming to identify all entities from a large corpus. Alternatively, biomedical entity taggers, such as ABNER (A Biomedical Named Entity Recognizer; Settles, 2005), can be employed to automate the entity-recognition process. These taggers can improve the efficiency at the cost of lower correctness of entity recognition.

### Relation Annotation

Statistical learning requires a training corpus of annotated relations from which patterns of true relations can be learned. To create such a training dataset, domain experts need to read the literature text in the corpora and manually annotate meaningful relations. For relation classification tasks, the type of biomedical relations, such as activation and inhibition, needs to be labeled as well. Some databases that summarize documented biomedical relations have been used as proxies for training data (Rosario & Hearst, 2005).

### Kernel Construction

Statistical learning requires both positive and negative examples. Hence, each pair of entities that appear in the

same sentence is regarded as a potential relation or a relation instance. Particularly, a sentence that contains multiple named entities can derive multiple relation instances. Each relation instance can be represented in a certain format: a bag-of-words, a word sequence, or a parse tree.

A bag-of-words is the simplest and most-used representation of relation instances. Specifically, given all words  $W = \{w_1, \dots, w_N\}$ , each relation instance is represented as a vector in which each element indicates the occurrence of a particular word in the sentence. In order to differentiate relation instances from the same sentence, the words that describe the two entities are not included in the word representation for each instance.

Furthermore, taking into account the sequential order of words in a sentence, we can represent a relation instance as a sequence of words,  $S = w_1 \dots w_N$ . For each relation instance, all words except for the two named entities are listed according to their sequence in the sentence. Such word and sequence representations have been used in existing studies such as (Lodhi et al., 2002) and (Bunescu et al., 2006). Both representations capture shallow lexical patterns as predictors for relation extraction. For a particular entity pair, all the other words (including other entity words) in the sentence, to some extent, form its lexical context. Hence, we keep all these words except the entity pair so as to insure that the contextual information is intact. By doing so, we also keep different relation instances from a sentence distinct from each other.

Given a sentence, a syntax parser can create a parse tree that represents the syntactic structure of the sentence according to some formal grammar (Lease & Charniak, 2005). A parse tree is made up of nodes connected by branches. In a parse tree, a node is either a root node, a branch node, or a leaf

TABLE 1. Sample POS tags in Penn treebank.

Bracket levels	POS tags	Descriptions
Clause level	S	simple declarative clause
Phrase level	NP	noun phrase
	PP	prepositional phrase
	VP	verb phrase
Word Level	CC	coordinating conjunction
	DT	determiner
	EX	existential there
	IN	preposition or subordinating conjunction
	JJ	adjective
	NN	noun
	VBD	verb, past tense
	VBN	verb, past participle
	VBP	verb, non-3rd person singular present
VBZ	verb, 3rd person singular present	

node. Each leaf node corresponds to a word. Each node in the tree is annotated with a part-of-speech (POS) tag. Table 1 lists some common POS tags with descriptions. A detailed manual can be found in the part-of-speech tagging guidelines for the Penn Treebank Project (<http://www.cis.upenn.edu/~treebank/>). A relation instance is often represented as a minimum subtree that contains the two entities (Zelenko et al., 2003).

Figure 2 shows an example of these three representations for the same sentence: “Mutant p53 genes increase IL-6 expression,” in which “p53” and “IL-6” are identified as two biomedical entities (genes). In this example, the minimum subtree of the two entities is the whole parse tree.

Each representation captures different types of contextual information. In this study, we use different kernel functions  $K(x,y) = \phi(x) \cdot \phi(y)$  to compute the similarity between relation instances for each representation. Specifically, we adopt three popular kernel functions, i.e., the word kernel, the sequence kernel (Lohbi et al., 2002), and the tree kernel

(Zelenko et al., 2003), for relation extraction. In particular, we propose a novel trace-tree kernel to overcome the limitations of the standard tree kernel. Moreover, we define several composite kernels by combining kernel functions so as to enhance the performance for relation extraction.

*Word Kernel.* In a bag-of-words representation, each relation instance  $x$  is represented as a vector  $\phi(x)$  indicating the occurrence of each word in the sentence. Here,  $\phi_i(x) = 1$  if word  $i$  occurs in the sentence. A word kernel function (or a bag-of-words kernel)  $K_W(x, y)$  is defined on such a word representation. Without listing all the possible words,  $K_W$  is defined as the inner product of vectors  $\phi(x)$  and  $\phi(y)$  and returns the number of words in common between two instances.

$$K_W(x, y) = \phi(x) \cdot \phi(y)$$

The word kernel is simple and efficient. However, neither the word sequence nor the sentence structure is captured by word kernels.

*Sequence kernel.* Unlike a bag-of-words, the word sequence representation of a relation instance,  $S = w_1 \dots w_N$ , takes into account the sequential order of words in a sentence. Such a sequence can be further decomposed into subsequences of  $n$ -grams. The string kernel proposed by Lodhi et al. (2002) was first used for text classification by analyzing the subsequences of letters or words. Recently, this has been extended and applied to relation extraction from text (Bunescu et al., 2006).

A *subsequence* is a finite sequence of words. For sequence  $s, t$ , we denote by  $|s|$  the length of the sequence  $s = s_1 \dots s_{|s|}$ , and by  $st$  the sequence obtained by concatenating the sequences of  $s$  and  $t$ . The sequence  $s[i : j]$  is the subsequence  $s_i \dots s_j$  of  $s$ . We say that  $u$  is a subsequence of  $s$ , if there exist

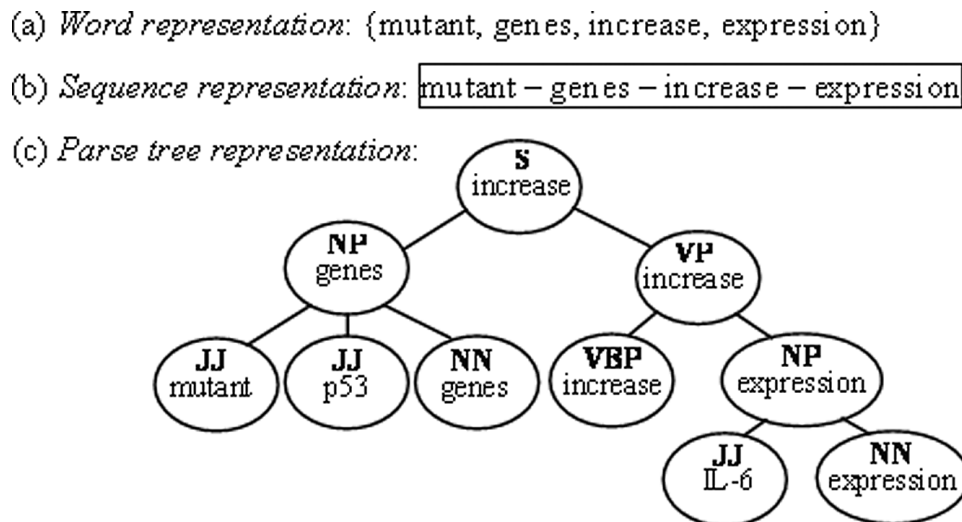


FIG. 2. An example of representations for a relation instance.

indices  $\mathbf{i} = (i_1, \dots, i_{|\mathbf{i}|})$ , with  $1 \leq i_1 < \dots < i_{|\mathbf{i}|} \leq |s|$ , such that  $u_j = s_{i_j}$  (the  $i_j$ th word in  $s$ ), for  $j = 1, \dots, |\mathbf{i}|$ , or  $u = s[\mathbf{i}]$  for short. The length  $l(\mathbf{i})$  of the subsequence in  $s$  is  $i_{|\mathbf{i}|} - i_1 + 1$ .

The feature mapping  $\phi$  for a sequence  $s$  is given by defining the  $u$  coordinate  $\phi_u(s)$  for each subsequence  $u$ . We define

$$\phi_u(s) = \sum_{i:u=s[\mathbf{i}]} \lambda^{l(\mathbf{i})}$$

where  $0 < \lambda \leq 1$ . These features measure the number of occurrences of subsequences in the  $s$ , weighting them according to their lengths.  $\lambda$  is the decay factor to penalize subsequences with more interior gaps and therefore longer length.

Hence, the inner product of the feature vectors for two sequences  $s$  and  $t$  give a sum over all common subsequences weighted according to their frequency of occurrence and length:

$$K_n(s, t) = \sum_{u \in \Sigma^n} \phi_u(s) \cdot \phi_u(t) = \sum_{u \in \Sigma^n} \sum_{i:u=s[\mathbf{i}]} \sum_{j:u=t[\mathbf{j}]} \lambda^{l(\mathbf{i})+l(\mathbf{j})}$$

The sequence kernel  $K_S$  accumulates the  $K_n$ s of different sequence length to get the overall similarity between two sequences:

$$K_S(s, t) = \sum_n K_n(s, t)$$

**Tree kernels.** Given a sentence, a syntax parser can create a parse tree that shows the syntactic structure. A tree (or a subtree)  $T$  is represented as  $\{p, [T.c]\}$ , where  $p$  is the  $T$ 's root node with a set of attributes  $V = \{v_1, v_2, \dots\}$  and  $[T.c]$  denotes  $p$ 's children (nodes or subtrees). The node attributes often consist of word, POS, and entity type (e.g., gene, protein, disease, or function). Some recent studies have incorporated attributes such as chunk-tag and Wordnet hypernyms to capture more semantic information (Culotta & Sorensen, 2004; Harabagiu, Bejan, & Morarescu, 2005). Given such structured representation, Zelenko et al. (2003) proposed tree kernels to extract relations (e.g., person-affiliation) from text. We introduce the standard tree kernel and our proposed trace-tree kernel.

**Standard tree kernel.** In order to focus on the most relevant information to relations, a standard tree kernel is often defined on the minimum subtree that contains both entities in a parse tree. Node attributes  $V = \{v_1, v_2, \dots\}$  and the structural information are used in the tree kernel definition. First, we need to define two functions over tree nodes: a matching function  $m(p_i, p_j) \in \{0, 1\}$  and a similarity function  $s(p_i, p_j) \in [0, \infty)$ . The matching function determines whether two nodes are matchable or not by comparing a subset of attributes,  $V^m \subseteq V$ :

$$m(p_i, p_j) = \begin{cases} 1, & \text{if } v_k^i = v_k^j, \forall v_k \in V^m \\ 0, & \text{otherwise} \end{cases}$$

where  $v_k^i$  and  $v_k^j$  is the value of attribute  $v_k$  of nodes  $p_i$  and  $p_j$ , respectively.

If two nodes are matchable, then the similarity function is computed by comparing the other attributes of nodes,  $V^s \subseteq V$ :

$$s(p_i, p_j) = \sum_{v_k \in V^s} \omega_k C(v_k^i, v_k^j)$$

where  $0 < \omega_k \leq 1$  is the weight of attribute  $k$  and  $C(v_k^i, v_k^j)$  is a function that computes the compatibility between two attribute values:

$$C(v_k^i, v_k^j) = \begin{cases} 1, & \text{if } v_k^i = v_k^j \\ 0, & \text{otherwise} \end{cases}$$

Then  $s(p_i, p_j)$  returns the weighted number of attributes values in common between  $p_i$  and  $p_j$ .

For two relation instances,  $T_1$  and  $T_2$ , we define the tree kernel  $K_T(T_1, T_2)$  that includes the similarity of the parent nodes and the similarity of the children:

$$K_T(T_1, T_2) = \begin{cases} 0, & \text{if } m(T_1.p, T_2.p) = 0 \\ s(T_1.p, T_2.p) + K_c(T_1.c, T_2.c), & \text{otherwise} \end{cases}$$

where the similarity function  $K_c$  is defined over children nodes  $T.c$ .

Let  $\mathbf{i}$  be a sequence of indices such that  $i_1 \leq i_2 \leq \dots \leq i_n$ , and likewise for  $\mathbf{j}$ . Let  $d(\mathbf{i}) = i_n - i_1 + 1$  and  $l(\mathbf{i})$  be the length of  $\mathbf{i}$ . For a relation instance  $T$ , let  $T[\mathbf{i}]$  denote a subsequence of children  $T.c = \{T[i_1], \dots, T[i_n]\}$ . Then we have

$$K_c(T_1.c, T_2.c) = \sum_{\mathbf{i}, \mathbf{j}, l(\mathbf{i})=l(\mathbf{j})} \lambda^{d(\mathbf{i})} \lambda^{d(\mathbf{j})} K(T_1[\mathbf{i}], T_2[\mathbf{j}])$$

where constant  $0 < \lambda \leq 1$  is a decay factor that decreases the similarity between two sequences that are spread out within children sequences. For a pair of matching instances  $T_1$  and  $T_2$  such that  $m(T_1.p, T_2.p) = 1$ , the kernel function  $K(T_1, T_2)$  needs to recursively compute the matching sequences of their children and accumulate the similarity scores. Tree kernels can be categorized into contiguous tree kernels and sparse tree kernels (Zelenko et al., 2003). A contiguous kernel only matches contiguous subsequences in children,  $d(\mathbf{i}) = l(\mathbf{i})$ , whereas a sparse tree kernel allows noncontiguous sequences,  $d(\mathbf{i}) \geq l(\mathbf{i})$ , which entails higher computational costs.

**Trace-tree kernel.** A relationship between two biomedical entities often follows one of the following three patterns (Bunescu et al., 2006; Fundel, Küffner, & Zimmer, 2007): (a) **[B] Between:** only words between the two entities express the relationship. Example: "X interacts with Y." (b) **[FB] Fore-Between:** words before and between the two entities simultaneously express the relationship. Example: "... interaction of X and Y ...". (c) **[BA] Between-After:**

words between and after the two entities simultaneously express the relationship. Example: “X and Y interact . . .”

In the standard tree kernel, a relation instance is the minimum subtree that contains the two entities. If the relationship follows pattern [B], this minimum subtree often can capture the most relevant information in a parse tree. Unfortunately, in cases of [FB] or [BA], since some relation-indicating elements appear before or after the two entities, the minimum subtree may lose such contextual information in the remainder of the parse tree. For instance, Figure 3 shows the parse tree of a sentence: “There is a functional interaction between WT1 and p53.” The minimum subtree contains limited contextual information, while some important relation indicators such as “interaction” and “between” are not captured.

In order to catch such information without extending the size the subtree, we modified the tree kernel function by incorporating an auxiliary kernel. Specifically, we only focus on the trace from the root node of the minimum subtree to the root of the full parse tree. Such a trace is a sequence of head words of the branch nodes. For the relation instance shown in Figure 3, the trace is the word sequence: p53–between–interaction–is–is. In a parse tree, the head word of a branch node indicates the key meaning of the corresponding subtree. Hence, such a trace captures more “global” context in the full parse tree in addition to the minimum subtree. Along this trace, branch nodes closer to the trace head (i.e., the root of minimum subtree) are assumed to be more relevant to the relation of interest. We imitate the sequence kernel and define a trace-kernel function to measure the similarity between traces.

For trace  $s$ , we denote by  $|s|$  the length of the trace  $s = s_1 \dots s_{|s|}$ . The sequence  $s[i : j]$  is the subsequence  $s_i \dots s_j$  of  $s$ . Like the sequence kernel, the trace kernel compares the

subsequences. By following the same notations, the feature mapping  $\phi$  for a trace  $s$  is given by defining the coordinate  $\phi_u(s)$  for each subsequence  $u$  of length  $n$ . We define:

$$\phi_u(s) = \sum_{i:u=s[i]} \lambda^{l(i)} \mu^i$$

where  $0 < \lambda, \mu \leq 1$ . These features measure the number of occurrences of subsequences in the trace  $s$ , weighting them according to their lengths  $l(i)$  and distances from the trace end ( $i$ ).  $\lambda$  is the decay factor to penalize subsequences with more interior gaps and therefore longer length;  $\mu$  is the decay factor to penalize subsequences starting farther from the trace head.

Hence, the inner product of the feature vectors for two traces  $s$  and  $t$  give a sum over all common subsequences weighted according to their frequency of occurrence, length, and position:

$$K_n(s, t) = \sum_{u \in \Sigma^n} \phi_u(s) \cdot \phi_u(t) = \sum_{u \in \Sigma^n} \sum_{i:u=s[i]} \sum_{j:u=t[j]} \lambda^{l(i)+l(j)} \mu^{i+j}$$

Following the method in Lodhi et al. (2002), we introduce an additional function to aid in defining a recursive computation for this kernel:

$$K'_i(s, t) = \sum_{u \in \Sigma^n} \sum_{i:u=s[i]} \sum_{j:u=t[j]} \lambda^{|s|+|t|-i-j+2} \mu^{i+j},$$

$$i = 1, \dots, n - 1$$

This function counts the length from the beginning of a particular sequence through to the end of the traces  $s$  and  $t$  instead of just  $l(i)$  and  $l(j)$ . We can now define a recursive computation for  $K'_i$  and hence compute  $K_n$ :

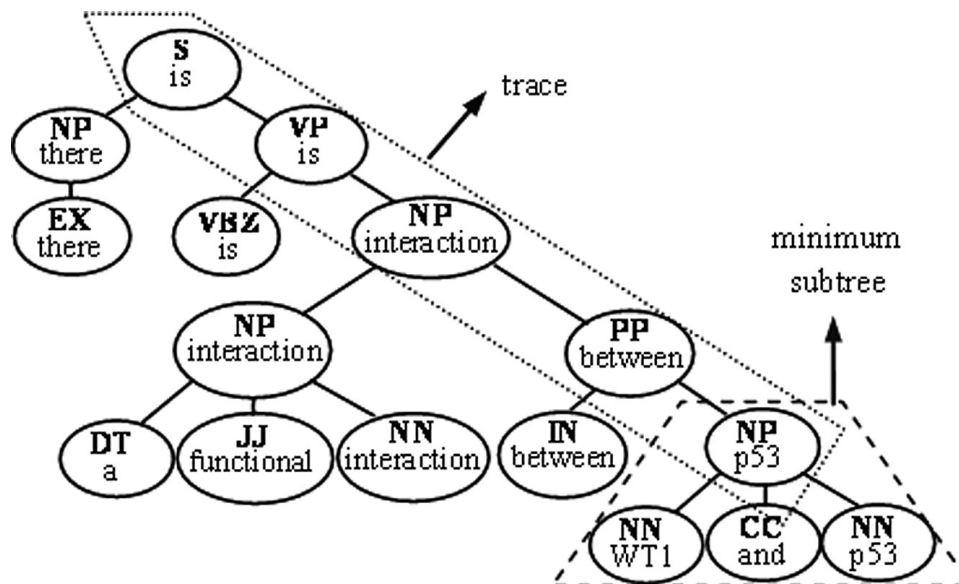


FIG. 3. An example of a minimum subtree and a trace in a parse tree.

$$K'_0(s, t) = \mu^{|s|+|t|+2}, \quad \text{for all } s, t,$$

$$K'_i(s, t) = 0, \quad \text{if } \min(|s|, |t|) < i,$$

$$K_i(s, t) = 0, \quad \text{if } \min(|s|, |t|) < i,$$

$$K'_i(sx, t) = \lambda K'_i(s, t) +$$

$$\sum_j \lambda^{|t|-j+2} K'_{i-1}(s, t[1:j-1]) \cdot c(x, t[j]), \quad i = 1, \dots, n-1,$$

$$K_n(sx, t) = K_n(s, t) + \sum_j \lambda^2 K'_{n-1}(s, t[1:j-1]) \cdot C(x, t[j]),$$

where  $c(x, y)$  is function that compares two words. It returns 1 if  $x$  equals  $y$  and 0 otherwise.

The computational efficiency can be further improved by introducing an additional function  $K''_i$ :

$$K'_i(sx, t) = \lambda K'_i(s, t) + K''_i(sx, t), \quad i = 1, \dots, n-1$$

$$K''_i(sx, ty) = \lambda K''_i(s, t) + \lambda^2 K'_{i-1}(s, t) \cdot c(x, y), \quad i = 1, \dots, n-1$$

Like the sequence kernel, the trace-kernel function  $K_R$  accumulates  $K_n$  of different sequence length to get the overall similarity between two traces:

$$K_R(s, t) = \sum_n K_n(s, t)$$

The trace-tree kernel  $K_{TR}$  combines the similarity between two subtrees and the similarity between two traces, i.e., a standard tree kernel  $K_T$  and a trace kernel  $K_R$ :

$$K_{TR}(x, y) = K_T(x, y) + K_R(x, y)$$

**Composite kernels.** The kernel functions above cast heterogeneous data representation into the common format of kernel matrices. This allows various kernel matrices to be combined into more complicated kernels using basic algebraic operations such as addition, multiplication, and exponentiation (Cristianini & Shawe-Taylor, 2000; Lanckriet et al., 2004). For example, given two kernels  $K_1$  and  $K_2$ , the combined kernel  $K(x, y) = \alpha_1 K_1(x, y) + \alpha_2 K_2(x, y)$  is also a valid kernel. Previous studies have shown that composite kernels may improve the performance of relation extraction (Culotta & Sorensen, 2004; Zhao & Grishman, 2005).

### Learning and Evaluation

After kernel computation is performed, relation instances from the corpus are transformed into a kernel matrix that contains the similarity scores between instances. In order to learn a classification model for relation extraction, we need to assign each relation instance in the training set with a class label. These class labels are derived from the results of relation annotation described above. Specifically, relation detection is a binary classification problem to predict whether a pair of entities has a relation or not. In this case, each relation instance is assigned with a label of “1/true” or “0/false,” where a true relation means

the sentence does indicate a relationship between the two entities. Furthermore, to classify different types of relations, each instance needs to be assigned with a label out of multiple possibilities, each representing a specific relation type. Next, a kernel machine such as SVM (Cristianini & Shawe-Taylor, 2000) can be applied to learn a classification model that can identify biomedical relations in future text. We can evaluate the performance of the relation-extraction model by comparing its prediction of relations on a test set with the relations’ true labels.

### Experimental Study

We conducted experiments to evaluate different kernel methods for relation extraction from biomedical literature.

#### Test Bed

In order to examine the kernel methods for biomedical relation extraction, we conducted experiments on a corpus of literature abstracts from Medline. In total, the corpus consisted of 19,000 cancer-related abstracts. Cancer has been chosen as our test bed because of its significance in biomedical research. There is a wide range of words and phrases that refer to types of cancer as well. The collection was identified by a query on PubMed with the following parameters: must contain PubMed links for NCBI gene that relates to the keywords “cancer\* OR neoplas\* OR tumor\*”; must have an abstract, which must be limited to Medline; and must have a MeSH term “neoplasms” (cancer) or its subtypes. This query was verified by a domain expert. We randomly selected 200 abstracts out of the 19,000 abstracts and conducted our experiments on this test bed, giving a similar scale to other studies (Bunescu et al., 2006; Marcotte, Xenarios, & Eisenberg, 2001). A biomedical scientist manually annotated biomedical entities such as genes, proteins, functions, and diseases. Table 2 summarizes the statistics of our test bed. Among the 1,815 sentences in the corpus, 154 contain no entity and 300 contain only one entity. Therefore, no candidate relation instance can be derived from these sentences. There were 450 sentences containing two entities and therefore one relation instance. The remaining 911 sentences contain at least three entities and can derive multiple relation instances.

Because our study only deals with intrasentence relations, relations can only be extracted from sentences that contain at least two entities. Each entity pair from the same sentence is regarded as a candidate relation instance. A sentence consisting

TABLE 2. Statistics of the biomedical abstract test bed.

Number of abstracts	200
Total number of sentences	1,815
Average number of sentences per abstract	9.08
Total number of entities	5,109
Average number of entities per sentence	2.81
Number of sentences that contain 0 entities	154
Number of sentences that contain 1 entity	300
Number of sentences that contain 2 entities	450
Number of sentences that contain $\geq 3$ entities	911

of multiple entities can derive multiple relation instances. In total, there are 8,071 relation instances are derived from our test bed of 200 abstracts. Based on the relation annotation by the biomedical scientist, 2,156 out of the 8,071 instances were identified as true relations, while the remaining 5,915 instances were labeled as false relations by default. According to the categorization defined by Marshall, Su, McDonald, Eggers, and Chen (2006), true relations were further divided into four relation types: induction, suppression, nondirectional association, and directional association. Table 3 summarizes the biomedical relations identified from our test bed.

### Evaluation Metrics

We use standard machine-learning evaluation metrics—accuracy, precision, recall, and F-measure—to evaluate the performances of relation extraction. These metrics have been widely used in information extraction and relation extraction studies (Zelenko et al., 2003; Bunescu et al., 2005).

In particular, accuracy measures the overall correctness of classification:

$$\text{accuracy} = \frac{\text{\# of all correctly identified instances}}{\text{total \# of instances}}$$

Precision, recall, and F-measure evaluate the correctness for each class. Specifically, precision indicates the correctness of identified relations, and recall indicates the completeness of identified relations. F-measure is the harmonic mean of precision and recall.

$$\text{precision}(i) = \frac{\text{\# of correctly identified instances for class } i}{\text{total \# of instances identified as class } i},$$

$$\text{recall}(i) = \frac{\text{\# of correctly identified instances for class } i}{\text{total \# of instances in class } i},$$

$$\text{F-measure}(i) = \frac{2 \times \text{precision}(i) \times \text{recall}(i)}{\text{precision}(i) + \text{recall}(i)}$$

### Hypotheses

Our study is aimed at testing the following three hypotheses:

1. A tree kernel will outperform a sequence kernel and a word kernel for biomedical relation extraction.

TABLE 3. Tagged biomedical relations from 200 abstracts.

Relation Types	Number	Percentage
true	2,156	26.71%
1. induction	489	6.06%
2. suppression	244	3.02%
3. nondirectional association	1,058	13.11%
4. directional association	365	4.52%
false	5,915	73.29%
Total	8,071	100.00%

2. The trace-tree kernel will outperform the standard tree kernel.
3. The composition of a word/sequence kernel and a tree kernel will outperform individual kernels alone.

### Experimental Design

In this study, we compare different kernel methods by performing two tasks: (a) relation detection: a binary classification of true and false relations, and (b) relation classification: a 4-class classification of the four relation types. All the 8,017 relation instances were used for the relation detection task, while the 2,156 true relations were used to perform the 4-class classification.

In addition to a word kernel ( $K_W$ ), a sequence kernel ( $K_S$ ), and a standard tree kernel ( $K_T$ ), we also implemented the trace-tree kernel  $K_{TR}$  and four composite kernels:  $K_{TW}$  ( $K_T + K_W$ ),  $K_{TS}$  ( $K_T + K_S$ ),  $K_{TRW}$  ( $K_{TR} + K_W$ ), and  $K_{TRS}$  ( $K_{TR} + K_S$ ).

To parse the sentences in our corpus, we used the parser developed by Lease and Charniak (2005), which was trained on biomedical corpora. Head words in branch nodes were assigned based on standard head-word propagation rules (Collins, 1997). In a parse tree, three node attributes are considered: word, POS, and type. In the computation of tree kernels,  $V^m = \{\text{POS, type}\}$  are used in the matching function, while in the similarity function  $V^s = \{\text{word}\}$  is used and weight  $\omega_k$  is set to 1 (Zelenko et al., 2003). We chose a continuous tree kernel due to its smaller computational cost. The decay factors  $\lambda$  and  $\mu$  were set to 0.5 for all kernels.

We chose an SVM as the kernel machine due to its excellent performance, reported in many applications. In particular, we chose an SVM package, LibSVM, for kernel learning ([www.csie.ntu.edu.tw/~cjlin/libsvm](http://www.csie.ntu.edu.tw/~cjlin/libsvm)), because (a) it has been frequently used in previous studies (Culotta & Sorensen, 2004), (b) it supports multiclass classification, (c) it accepts customized kernels, and (d) it performs parameter selection for better performance.

We evaluated the eight kernel methods for both relation detection and classification on our test bed. For each kernel, we used 90% of the relation instances as the training set to learn a classification model, and predicted the class labels of the remaining 10% of instances for the testing set. We repeated this process thirty times by randomly splitting the datasets for statistical analysis.

TABLE 4. Kernel methods for biomedical relation detection.

Types	Kernels	Accuracy	Precision	Recall	F-measure
Word	$K_W$	73.10%	49.67%	32.33%	39.09%
Sequence	$K_S$	78.02%	60.72%	50.69%	55.21%
Tree	$K_T$	79.13%	62.19%	56.56%	59.16%
	$K_{TR}$	80.52%	64.64%	60.22%	62.31%
Composite	$K_{TW}$	80.21%	63.91%	60.15%	61.92%
	$K_{TS}$	81.97%	67.54%	63.06%	65.17%
	$K_{TRW}$	81.61%	67.12%	61.62%	64.21%
	$K_{TRS}$	<b>83.14%</b>	<b>70.11%</b>	<b>64.68%</b>	<b>67.23%</b>

TABLE 5. Kernel methods for biomedical relation classification.

Types	Kernels	Accuracy	Class	Precision	Recall	F-measure
Word	$K_W$	66.42%	1	64.09%	59.72%	61.63%
			2	53.63%	45.20%	48.55%
			3	71.15%	77.77%	74.22%
			4	62.99%	56.94%	59.56%
			Average	62.97%	59.91%	60.99%
Sequence	$K_S$	72.67%	1	70.42%	<b>66.74%</b>	68.33%
			2	59.99%	49.20%	53.69%
			3	<b>75.53%</b>	84.65%	<b>79.78%</b>
			4	75.11%	61.89%	67.42%
			Average	70.26%	65.62%	67.30%
Tree	$K_T$	69.09%	1	66.43%	56.32%	60.77%
			2	68.09%	44.93%	53.41%
			3	67.95%	85.47%	75.66%
			4	82.05%	55.05%	65.48%
			Average	71.13%	60.44%	63.83%
	$K_{TR}$	69.18%	1	65.88%	57.43%	61.06%
			2	65.31%	48.00%	54.68%
			3	69.21%	83.30%	75.56%
			4	78.07%	58.29%	66.51%
			Average	69.62%	61.76%	64.46%
Composite	$K_{TW}$	71.16%	1	67.78%	62.99%	65.10%
			2	67.74%	52.93%	58.97%
			3	71.73%	84.37%	77.51%
			4	78.32%	56.22%	65.08%
			Average	71.39%	64.13%	66.67%
	$K_{TS}$	73.60%	1	70.25%	65.35%	67.58%
			2	<b>69.30%</b>	50.67%	58.07%
			3	73.62%	<b>85.63%</b>	79.12%
			4	<b>82.47%</b>	65.32%	72.58%
			Average	73.91%	66.74%	69.34%
	$K_{TRW}$	70.76%	1	68.82%	64.79%	66.49%
			2	62.46%	53.33%	56.98%
			3	72.43%	80.35%	76.14%
			4	74.22%	62.79%	67.81%
			Average	69.48%	65.32%	66.86%
$K_{TRS}$	<b>74.20%</b>	1	<b>71.56%</b>	65.76%	<b>68.36%</b>	
		2	67.51%	<b>58.00%</b>	<b>61.95%</b>	
		3	74.88%	84.59%	79.38%	
		4	81.75%	<b>66.31%</b>	<b>72.92%</b>	
		Average	<b>73.93%</b>	<b>68.67%</b>	<b>70.65%</b>	

### Results and Discussion

The performance measures of the eight kernel methods for relation detection and classification are summarized in Tables 4 and 5, respectively. Specifically, since relation detection is aimed at identifying the true relations from all candidate instances, we focus only on the precision, recall, and F-measure for the class of true relations. For relation classification, we report both the individual precision, recall, and F-measure values for each class and the average values of these measures. The values in bold fonts are the best performances among the eight learning methods.

In our experiments, for relation detection and relation classification, the composite kernel  $K_{TRS}$  achieved the best performance among all kernels. For relation detection,  $K_{TRS}$  achieved 83.14% accuracy, 70.11% precision, 64.68% recall, and 67.23% F-measure. For relation classification,  $K_{TRS}$  achieved 74.20% accuracy, 73.93% average precision, 68.67% average recall, and 70.65% average F-measure. Specifically,  $K_{TRS}$  achieved the highest F-measure scores for relation classes 1, 2, and 4: F-measure(1) = 68.36%, F-measure(2) = 61.95%, and F-measure(4) = 72.92%. For class 3,  $K_{TRS}$  achieved the second highest F-measure—79.38%, slightly lower than 79.78%, achieved by  $K_S$ .

TABLE 6. Hypothesis testing for biomedical relation detection.

Hypothesis No.	Hypothesis	Accuracy	Precision	Recall	F-measure
1	$K_S > K_W$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_T > K_W$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_T > K_S$	0.0005**	0.0304*	<0.0001**	<0.0001**
2	$K_{TR} > K_T$	<0.0001**	0.0002**	0.0001**	<0.0001**
3	$K_{TW} > K_W$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_{TW} > K_T$	0.0003**	0.0080**	<0.0001**	<0.0001**
	$K_{TS} > K_S$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_{TS} > K_T$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_{TRW} > K_W$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_{TRW} > K_{TR}$	0.0006**	0.0005**	0.0417*	0.0018**
	$K_{TRS} > K_S$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
$K_{TRS} > K_{TR}$	<0.0001**	<0.0001**	<0.0001**	<0.0001**	

Note. Significance levels \* $\alpha = 0.05$  and \*\* $\alpha = 0.01$ .

TABLE 7. Hypothesis testing for biomedical relation classification.

Hypothesis No.	Hypothesis	Accuracy	Precision	Recall	F-measure
1	$K_S > K_W$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_T > K_W$	0.0008**	<0.0001**	0.2899	0.0018**
	$K_T > K_S$	<0.0001**	0.2109	<0.0001**	0.0006**
2	$K_{TR} > K_T$	0.4556	0.0969	0.1053	0.2768
3	$K_{TW} > K_W$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_{TW} > K_T$	0.0008**	0.3816	<0.0001**	0.0003**
	$K_{TS} > K_S$	0.1269	0.0007**	0.1518	0.0303*
	$K_{TS} > K_T$	<0.0001**	0.0048**	<0.0001**	<0.0001**
	$K_{TRW} > K_W$	<0.0001**	<0.0001**	<0.0001**	<0.0001**
	$K_{TRW} > K_{TR}$	0.0340*	0.4526	0.0010**	0.0158*
	$K_{TRS} > K_S$	0.0390*	0.0005**	0.0040**	0.0013**
$K_{TRS} > K_{TR}$	<0.0001**	0.0002**	<0.0001**	<0.0001**	

Note. Significance levels\* $\alpha = 0.05$  and \*\* $\alpha = 0.01$ .

Furthermore, in order to test our three hypotheses, we conducted pairwise single-sided  $t$  tests on the four evaluation metrics: accuracy, average precision, average recall, and average F-measure. The  $p$  values for the tests of our hypotheses for relation detection and classification are presented in Tables 6 and 7, where  $p$  values with \* and \*\* indicate significant difference at the level of  $\alpha = 0.05$  and  $0.01$ , respectively. The underlined  $p$  values indicate that the results contradict the hypotheses. Overall, most of our hypotheses were supported by our experiments.

**Hypothesis 1:  $K_T > K_S > K_W$ .** For both relation detection and classification, the sequence kernel  $K_S$  significantly outperformed the word kernel  $K_W$  for all four metrics with  $p$  values less than 0.0001. This is because a sequence kernel captures not only the common words in two relation instances but also the linear sequences of words in sentences. The linear sequences are thus shown to provide richer contextual information than a bag-of-words.

For relation detection, the standard tree kernel  $K_T$  was shown to significantly outperform both  $K_W$  and  $K_S$  with most  $p$  values less than 0.01, except for the precision of  $K_S < K_T$

( $p = 0.0304 < 0.05$ ). This is attributed to tree kernels' ability to capture the structural patterns between entities in a parse tree. Furthermore, in long sentences with multiple entities and complex structures, relations tend to exist between entities close to each other. Since  $K_W$  and  $K_S$  require comparing all the words or word subsequences in sentences, they may include some noise, which may reduce the extraction performances. Unlike  $K_W$  and  $K_S$ ,  $K_T$  is defined on the minimum subtree that covers the local context of the two entities and therefore excludes the noise from other parts of the tree. For instance, in the sentence "In contrast to the potent induction of IP-10 by IFN, . . .", the phrase "the potent induction of IP-10 by IFN" indicates an induction relation between "IP-10" and "IFN." Both  $K_W$  and  $K_S$  failed to identify this relation due to the noise from the rest of the sentence. In contrast, by focusing on the minimum subtree only, as shown in Figure 4,  $K_T$  filtered out the noise and identified the relation.

For relation classification,  $K_T$  also outperformed  $K_W$  significantly except for recall ( $p = 0.2899$ ). However, the hypothesis that  $K_T$  outperforms  $K_S$  for relation classification was not supported. In contrast, our experiments showed that  $K_S$  significantly outperformed  $K_T$  on accuracy, recall, and

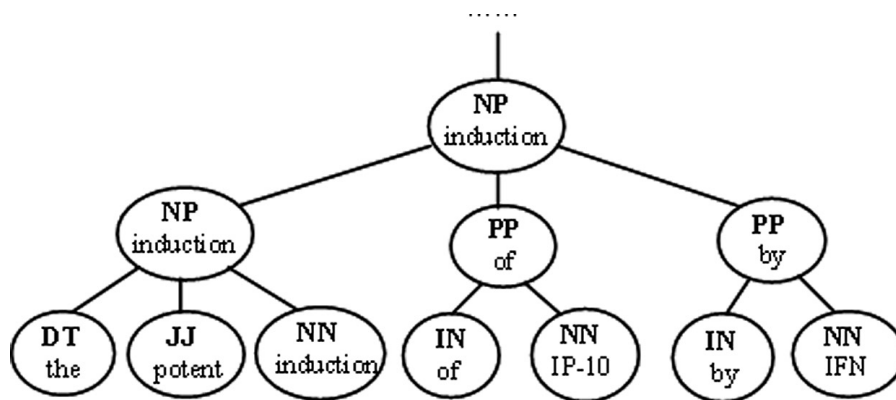


FIG. 4. The minimum subtree of entity pair “IP-10” and “IFN.”

F-measure at the level of  $\alpha = 0.01$ . The tree kernel captures rich structural information in a subtree. Such information is effective for differentiating true relations from false ones. However, given a true relation, the tree structure seems not as effective as lexical patterns (word sequences) for distinguishing different types of relations.

In summary, the parse-tree representation captures richer structural information, which can help identify relations but did not show significant benefits for relation classification as compared to the sequence representation.

*Hypothesis 2:  $K_{TR} > K_T$ .* For relation detection, compared with the standard tree kernel  $K_T$ , our proposed trace-tree kernel  $K_{TR}$  improved the accuracy from 79.13% to 80.52%, precision from 62.19% to 64.64%, recall from 56.56% to 60.22%, and F-measure from 59.16% to 62.31%, as shown in Table 4. Furthermore, statistical testing reported in Table 6 showed that all these improvements were significant ( $p < 0.01$ ). This confirmed our second hypothesis. For the standard tree kernel  $K_T$ , while minimum subtrees exclude noise in sentences, they may also leave out some relevant information for relation extraction. Such “short” subtrees in the training set may introduce much ambiguity into the learning process. The classification model trained on such data may not learn informative patterns for making accurate predictions. Without exploring the whole parse tree, the trace kernel captures the main contextual information missing in a minimum subtree and therefore it can significantly improve the relation-detection performance. For instance, the sentence “The IL-8 mRNA was induced by IL-1 $\beta$  and TNF $\alpha$ ” indicates two induction relations: “IL8–IL-1 $\beta$ ” and “IL8–TNF $\alpha$ ,” but not a relation between the entity pair, “IL-1 $\beta$ ” and “TNF $\alpha$ .” As shown in Figure 5, in the whole parse tree, the minimum subtree of “IL-1 $\beta$ ” and “TNF $\alpha$ ” only covers the phrase “IL-1 $\beta$  and TNF $\alpha$ .” Because this relation instance shares the same subtree structure (“A and B”) as true relations (e.g., our example “WT1 and p53” in the section on trace-tree kernel construction, above),  $K_T$  mistakenly identified this entity pair as a relation. By contrast, by incorporating the trace TNF $\alpha$  – by – induced – was – was into the kernel computation, the trace-tree

kernel  $K_{TR}$  identified that this sentence did not indicate a relation between “IL-1 $\beta$ ” and “TNF $\alpha$ .” Instead,  $K_{TR}$  successfully detected the two true relations: “IL8 – IL-1 $\beta$ ” and “IL8 – TNF $\alpha$ .”

This example shows how trace kernels can capture additional contextual information to improve relation detection. However, in our 200-abstract test bed, sentences of such a form are relatively rare. Moreover, 911 out of 1,815 sentences in our test bed contain at least three entities and therefore multiple entity pairs. Unlike our example, in which two out of three entity pairs are true relations, most multientity sentences derive a lot more false relations than true relations. The skewed distribution of positive (26.71%) and negative (73.29%) examples further increases the difficulty of training a classification model. Hence, the absolute values of improvement in relation-extraction performances seemed limited. In addition, as shown in our results, structural information is not as effective for relation classification as for relation detection. This explains why, as compared with  $K_T$ ,  $K_{TR}$  did not improve the performance of relation classification ( $p > 0.05$  for all four measures).

*Hypothesis 3: A composite kernel will outperform individual kernels.* Kernel functions convert different structural representations into the same format (a kernel matrix), which enables the composition of multiple kernels. By integrating different types of information, composite kernels can improve the performance of relation detection and classification. In our experiments, for relation detection, except that the recall of  $K_{TRW}$  was significantly higher than that of  $K_{TR}$  at  $\alpha = 0.05$ , all the other comparisons were consistent with our hypothesis and showed significant differences at the level of  $p$  less than 0.01. For relation classification, we observed that in most cases, composite kernels ( $K_{TW}$ ,  $K_{TS}$ ,  $K_{TRW}$ , and  $K_{TRS}$ ) outperformed their subkernels at the significance level  $\alpha = 0.05$  or  $\alpha = 0.01$ . We only found four comparisons that did not show significant difference:  $K_{TW} > K_T$  in precision ( $p = 0.3816$ ),  $K_{TS} > K_S$  in accuracy ( $p = 0.1269$ ) and recall ( $p = 0.1518$ ), and  $K_{TRW} > K_{TR}$  in precision (contradictory to

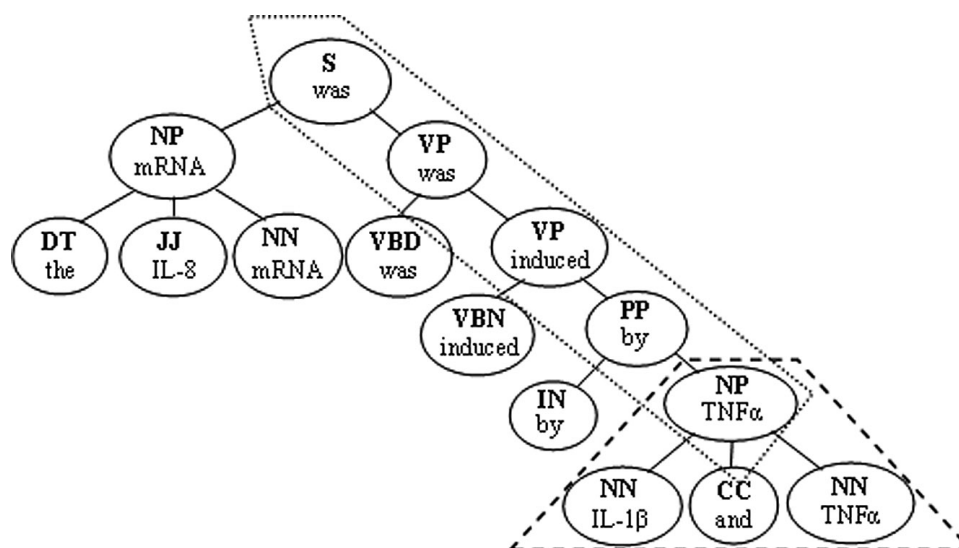


FIG. 5. The minimum subtree and trace of entity pair “IL-1 $\beta$ ” and “TNF $\alpha$ .”

the hypothesis but  $p = 0.4526$ ). In these four cases, the composite kernels achieved performances comparable to the subkernels in certain evaluation measures. Notably, their average F-measures of composite kernels were all significantly higher than that of the subkernels.

## Conclusions and Future Directions

Automatically extracting relations from text is an important and challenging task in NLP. Recently, relation extraction from biomedical literature corpora has attracted substantial attention due to the rapid growth of biomedical research development and discovery. In this study, we developed a framework of kernel-based learning methods for biomedical relation extraction. In particular, we proposed a novel trace-tree kernel that extends a standard tree kernel by adding a trace kernel to capture richer contextual information. We conducted an experimental study to compare different kernel methods on a corpus of biomedical literature abstracts. The results showed excellent performance of the tree kernel for relation extraction. Furthermore, compared with the tree kernel, our proposed trace-tree kernel significantly improved the relation-extraction performance. In addition, the good performance of composite kernels demonstrated the effectiveness of integrating contextual information represented by different data types.

We are in the process of extending our study in the following directions: (a) Kernel functions in most studies that compare words are based on exact match. We plan to redesign kernels that can capture semantic similarity between words so as to improve the performance of relation extraction. (b) In our study, named entities have been manually tagged by domain experts before relation extraction. We will incorporate an automated named-entity recognition module

into our relation-extraction framework and reexamine the overall performance. (c) Because domains may vary in the manner of describing relations, a classification model trained on a corpus of a specific domain may not perform well on other domains. However, the framework of kernel-based learning for relation extraction is generally applicable to not only biomedicine but also other domains. In our future research, we plan to examine these kernel methods for relation extraction in application areas such as online forums and Weblog analysis.

## Acknowledgments

The project is supported by an NIH/NLM grant, R33 LM07299-01, 2002-2005, “Genescene: a Toolkit for Gene Pathway Analysis.”

## References

- Bunescu, R., Ge, R., Kate, R.J., Marcotte, E.M., Mooney, R.J., Ramani, A.K., et al. (2005). Comparative experiments on learning information extractors for proteins and their interactions. *Artificial Intelligence in Medicine*, 33(2), 139–155.
- Bunescu, R., & Mooney, R. (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP)*. (pp. 724–731). Morristown, NJ: ACL.
- Bunescu, R., Mooney, R., Weiss, Y., Schölkopf, B., & Platt, J. (2006). Subsequence kernels for relation extraction. *Advances in Neural Information Processing Systems*, 18, 171–178. Retrieved January 1, 2007, from [http://books.nips.cc/papers/files/nips18/NIPS2005\\_0450.pdf](http://books.nips.cc/papers/files/nips18/NIPS2005_0450.pdf)
- Chen, W.S., Yuen, P.C., Huang, J., & Dai, D.Q. (2005). Kernel machine-based one-parameter regularized Fisher discriminant method for face recognition. *IEEE Transactions on Systems Man and Cybernetics, Part B: Cybernetics*, 35(4), 659–669.
- Collins, M. (1997). Three generative, lexicalized models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics* (pp. 16–23). Morristown, NJ: ACL.

- Collins, M., & Duffy, N. (2001). Convolution Kernels for Natural Language. *Advances in Neural Information Processing Systems*, 14. Retrieved January 1, 2007, from <http://books.nips.cc/papers/files/nips14/AA58.pdf>
- Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods. New York: Cambridge University Press.
- Culotta, A., & Sorensen, J. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)* (pp. 423–429). Morristown, NJ: ACL.
- Donaldson, I., Martin, J., de Bruijn, B., Wolting, C., Lay, V., Tuekam, B., et al. (2003). PreBIND and textomy: Mining the biomedical literature for protein-protein interactions using a support vector machine. *BMC Bioinformatics*, 4, Article 11. Retrieved January 1, 2007, from <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=153503>
- Friedman, C., Kra, P., Yu, H., Krauthammer, M., & Rzhetsky, A. (2001). GENIES: A natural language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17(Suppl. 1), S74–S82.
- Fundel, K., Küffner, R., & Zimmer, R. (2007). RelEx: Relation extraction using dependency parse trees. *Bioinformatics*, 23(3), 365–371.
- Gaizauskas, R., Demetriou, G., Artymiuk, P.J., & Willett, P. (2003). Protein structures and information extraction from biological texts: the PASTA System. *Bioinformatics*, 19(1), 135–143.
- Harabagiu, S.M., Bejan, C.A., & Morarescu, P. (2005). Shallow Semantics for Relation Extraction. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)* (pp. 1061–1066). Retrieved January 1, 2007, from <http://www.ijcai.org/papers/1589.pdf>
- Haussler, D. (1999). Convolution Kernels on Discrete Structures (Tech. Rep. UCS-CRL-99-10). University of California at Santa Cruz. Retrieved January 1, 2007, from <http://citeseer.ist.psu.edu/haussler99convolution.html>
- Hirschman, L., Yeh, A., Blaschke, C., & Valencia, A. (2005). Overview of BioCreAtIvE: Critical assessment of information extraction for biology. *BMC Bioinformatics*, 6(Suppl. 1). Retrieved January 1, 2007, from <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1869002>
- Jenssen, T.K., Laegreid, A., Komorowski, J., & Hovig, E. (2001). A literature network of human genes for high-throughput analysis of gene expression. *Nature Genetics*, 28(1), 21–28.
- Joachims, T., Cristianini, N., & Shawe-Taylor, J. (2001). Composite kernels for hypertext categorisation. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*. (pp. 250–257). San Francisco: Morgan Kaufmann.
- Lanckriet, G.R., De Bie, T., Cristianini, N., Jordan, M.I., & Noble, W.S. (2004). A statistical framework for genomic data fusion. *Bioinformatics*, 20(16), 2626–2635.
- Lease, M., & Charniak, E. (2005). Parsing biomedical literature. In R. Dale, K.-F. Wong, J. Su, & O.Y. Kwong (Eds.), *Lecture Notes in Computer Science: Vol. 3651. Proceedings of the Second International Joint Conference on Natural Language Processing (IJCNLP)*. (pp. 58–69). Berlin: Springer-Verlag.
- Leroy, G., Chen, H.C., & Martinez, J.D. (2003). A shallow parser based on closed-class words to capture relations in biomedical text. *Journal of Biomedical Informatics*, 36(3), 145–158.
- Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research*, 2(3), 419–444.
- Marcotte, E.M., Xenarios, I., & Eisenberg, D. (2001). Mining literature for protein-protein interactions. *Bioinformatics*, 17(4), 359–363.
- Marshall, B., Su, H., McDonald, D., Eggers, S., & Chen, H. (2006). Aggregating automatically extracted regulatory pathway relations. *IEEE Transactions on Information Technology in Biomedicine*, 10(1), 100–108.
- McDonald, D., Chen, H., Su, H., & Byron, M. (2004). Extracting gene pathway relations using a hybrid grammar: The Arizona relation parser. *Bioinformatics*, 20(18), 3370–3378.
- Mitsumori, T., Murata, M., Fukuda, Y., Doi, K., & Doi, H. (2006). Extracting protein-protein interaction information from biomedical text with SVM. *IEICE Transactions on Information and Systems*, E89D(8), 2464–2466.
- Novichkova, S., Egorov, S., & Daraselia, N. (2003). MedScan, a natural language processing engine for MEDLINE Abstracts. *Bioinformatics*, 19(13), 1699–1706.
- Park, J.C., Kim, H.S., & Kim, J.J. (2001). Bidirectional incremental parsing for automatic pathway identification with combinatory categorial grammar. *Pacific Symposium on Biocomputing*, Vol. 6 (pp. 396–407). Retrieved January 1, 2007, from <http://psb.stanford.edu/psb-online/proceedings/psb01/jcpark.pdf>
- Pustejovsky, J., Castaño, J., Zhang, J., Kotecki, M., & Cochran, B. (2002). Robust relational parsing over biomedical literature: Extracting inhibit relations. *Pacific Symposium on Biocomputing*, Vol. 7 (pp. 362–373). Retrieved January 1, 2007, from <http://psb.stanford.edu/psb-online/proceedings/psb02/pustejovsk.pdf>
- Rindflesch, T.C., Tanabe, L., Weinstein, J.N., & Hunter, L. (2000). EDGAR: Extraction of drugs, genes, and relations from the biomedical literature. *Pacific Symposium on Biocomputing*, Vol. 5 (pp. 514–525). Retrieved January 1, 2007, from <http://psb.stanford.edu/psb-online/proceedings/psb00/rindflesch.pdf>
- Rosario, B., & Hearst, M. (2005). Multi-way relation classification: Application to protein-protein interactions. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP)*. (pp. 732–739). Morristown, NJ: ACL.
- Settles, B. (2005). ABNER: An open source tool for automatically tagging genes, proteins, and other entity names in text. *Bioinformatics*, 21(14), 3191–3192.
- Shatkay, H., & Feldman, R. (2003). Mining the biomedical literature in the genomic era: An overview. *Journal of Computational Biology*, 10(6), 821–855.
- Stapley, B.J., & Benoit, G. (2000). Bibliometrics: Information retrieval visualization from co-occurrences of gene names in MEDLINE Abstracts. *Pacific Symposium on Biocomputing*, Vol. 5 (pp. 526–537). Retrieved January 1, 2007, from <http://psb.stanford.edu/psb-online/proceedings/psb00/stapley.pdf>
- Sun, A., Lim, E.-P., Ng, W.-K., & Srivastava, J. (2004). Blocking reduction strategies in hierarchical text classification. *IEEE Transactions on Knowledge and Data Engineering*, 16(10), 1305–1308.
- Thomas, J., Milward, D., Ouzounis, C., Pulman, S., & Carroll, M. (2000). Automatic extraction of protein interactions from scientific abstracts. *Pacific Symposium on Biocomputing*, Vol. 5 (pp. 538–549). Retrieved January 1, 2007, from <http://psb.stanford.edu/psb-online/proceedings/psb00/thomas.pdf>
- Yakushiji, A., Tateisi, Y., Miyao, Y., & Tsujii, J. (2001). Event extraction from biomedical papers using a full parser. *Pacific Symposium on Biocomputing*, Vol. 6 (pp. 408–419). Retrieved January 1, 2007, from <http://psb.stanford.edu/psb-online/proceedings/psb01/yakushiji.pdf>
- Yu, H., Han, J., & Chang, K.C.-C. (2004). PEBL: Web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering*, 16(1), 70–81.
- Zelenko, D., Aone, C., & Richardella, A. (2003). Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3(6), 1083–1106.
- Zhao, H.T., Yuen, P.C., & Kwok, J.T. (2006). A novel incremental principal component analysis and its application for face recognition. *IEEE Transactions on Systems Man and Cybernetics, Part B: Cybernetics*, 36(4), 873–886.
- Zhao, S., & Grishman, R. (2005). Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)* (pp. 419–426). New Brunswick, NJ: ACL.
- Zhou, S., & Wang, K. (2005). Localization site prediction for membrane proteins by integrating rule and SVM classification. *IEEE Transactions on Knowledge and Data Engineering*, 17(12), 1694–1705.