# Improving Semi-Supervised Acquisition of Relation Extraction Patterns

**Mark A. Greenwood and Mark Stevenson**
Department of Computer Science
University of Sheffield
Sheffield, S1 4DP, UK
{m.greenwood,marks}@dcs.shef.ac.uk

## Abstract

This paper presents a novel approach to the semi-supervised learning of Information Extraction patterns. The method makes use of more complex patterns than previous approaches and determines their similarity using a measure inspired by recent work using kernel methods (Culotta and Sorensen, 2004). Experiments show that the proposed similarity measure outperforms a previously reported measure based on cosine similarity when used to perform binary relation extraction.

## 1 Introduction

A recent approach to Information Extraction (IE) is to make use of machine learning algorithms which allow systems to be rapidly developed or adapted to new extraction problems. This reduces the need for manual development which is a major bottleneck in the development of IE technologies and can be extremely time consuming (e.g. Riloff (1996)).

A number of machine learning approaches have recently been applied. One is the use of iterative learning algorithms to infer extraction patterns from a small number of seed examples (Yangarber et al., 2000; Stevenson and Greenwood, 2005). These approaches use dependency analysis as the basis of IE patterns. Training text is parsed and a set of candidate patterns extracted. These patterns are then compared against the seeds with the most similar being selected and added to the seed set. (Optionally, a user may verify the patterns at this point.) The process is then repeated with the remaining patterns being compared to the enlarged seed set. The process continues until a suitable set of patterns has been learned. These approaches require only a small number of example extraction patterns which greatly reduces the effort required to develop IE systems.

While it has been found that these approaches are capable of learning useful IE patterns from a handful of examples (Yangarber et al., 2000; Stevenson and Greenwood, 2005) they are limited by the use of basic extraction patterns: SVO tuples. The patterns used by these systems are defined as a verb and its direct subject and/or object. They could then only extract a limited set of relations; those expressed using a verb and its direct arguments. For example, these patterns could identify the relation between *Jones* and *Smith* in the sentence *"Jones replaced Smith"*. However, no pattern consisting of a verb and its arguments could be constructed which could identify the same relation in *"Jones was named as Smith's successor."*

Others have suggested alternative approaches for generating extraction patterns from dependency trees, each of which allows a particular part of the dependency analysis to act as an extraction pattern. For example, Sudo et al. (2003) used patterns consisting of a path from a verb to any of its descendents (direct or indirect) while Bunescu and Mooney (2005) suggest the shortest path between the items being related. However, iterative learning algorithms, such as the ones used by Yangarber et al. (2000) and Stevenson and Greenwood (2005), have not made use of these more complex extraction patterns. Part of the reason for this is that these algorithms require a way of determining the similarity between patterns (in order to compare candidate patterns with the seeds). This process is straightforward for simple patterns, based on SVO tuples, but less so for more complex extraction patterns.

In this paper we present a semi-supervised algorithm for the iterative learning of relation extraction patterns which makes use of a more complex pattern representation than has been previously used by these approaches (Sections 2 and 3). The algorithm makes use of a similarity function based on those which have been proposed for use with non-iterative learning algorithms (Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005). These are extended to include information about lexical similarity derived from WordNet (Section 4). We present results of using patterns acquired through this similarity function to perform binary relation extraction (Sections 5 and 6).

## 2 Semi-Supervised Learning of Extraction Patterns

We begin by outlining the general process of learning extraction patterns using a semi-supervised algorithm, similar to one presented by Yangarber (2003).

1. For a given IE scenario we assume the existence of a set of documents against which the system can be trained. The documents are unannotated and may be either relevant (contain the description of an event relevant to the scenario) or irrelevant.
2. This corpus is pre-processed to generate a set of all patterns which could be used to represent sentences contained in the corpus, call this set $P$. The aim of the learning process is to identify the subset of $P$ representing patterns which are relevant to the IE scenario.
3. The user provides a small set of seed patterns, $P_{seed}$, which are relevant to the scenario. These patterns are used to form the set of currently accepted patterns, $P_{acc}$, so $P_{acc} \leftarrow P_{seed}$. The remaining patterns are treated as candidates for inclusion in the accepted set, these form the set $P_{cand}(= P - P_{acc})$.
4. A function, $f$, is used to assign a score to each pattern in $P_{cand}$ based on those which are currently in $P_{acc}$. This function assigns a real number to candidate patterns so $\forall\ c \in P_{cand},\ f(c, P_{acc}) \mapsto \mathbb{R}$. A set of high scoring patterns (based on absolute scores or ranks after the set of patterns has been ordered by scores) are chosen as being suitable for inclusion in the set of accepted patterns. These form the set $P_{learn}$.
5. (Optional) The patterns in $P_{learn}$ may be reviewed by a user who may remove any they do not believe to be useful for the scenario.
6. The patterns in $P_{learn}$ are added to $P_{acc}$ and removed from $P_{cand}$, so $P_{acc} \leftarrow P_{acc} \cup P_{learn}$ and $P_{cand} \leftarrow P_{acc} - P_{learn}$
7. Stop if an acceptable set of patterns has been learned, otherwise goto step 4

Previous algorithms which use this approach include those described by Yangarber et al. (2000) and Stevenson and Greenwood (2005). A key choice in the development of an algorithm using this approach is the process of ranking candidate patterns (step 4) since this determines the patterns which will be learned at each iteration. Yangarber et al. (2000) chose an approach motivated by the assumption that documents containing a large number of patterns already identified as relevant to a particular IE scenario are likely to contain further relevant patterns. This approach operates by associating confidence scores with patterns and relevance scores with documents. Initially seed patterns are given a maximum confidence score of 1 and all others a 0 score. Each document is given a relevance score based on the patterns which occur within it. Candidate patterns are ranked according to the proportion of relevant and irrelevant documents in which they occur, those found in relevant documents far more than in irrelevant ones are ranked highly. After new patterns have been accepted all patterns' confidence scores are updated, based on the documents in which they occur, and documents' relevance according to the accepted patterns they contain.

Stevenson and Greenwood (2005) suggested an alternative method for ranking the candidate patterns. Their approach relied on the assumption that useful patterns will have similar meanings to the patterns which have already been accepted. They chose to represent each pattern as a vector consisting of the lexical items which formed the pattern and used a version of the cosine metric to determine the similarity between pairs of patterns, consequently this approach is referred to as "cosine similarity". The metric used by this approach incorporated information from WordNet and assigned high similarity scores to patterns with similar meanings expressed in different ways.

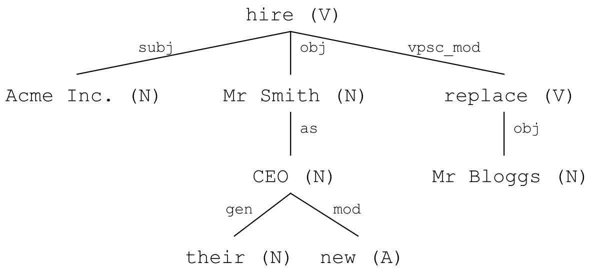

Figure 1: An example dependency tree.

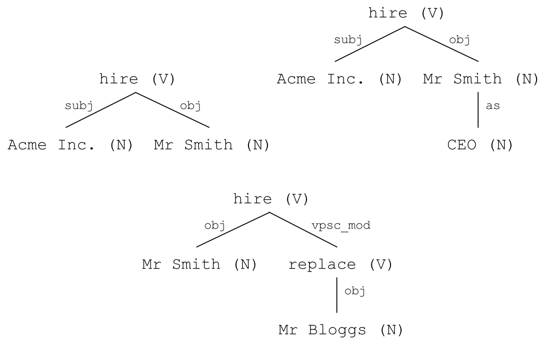

Figure 2: Example linked chain patterns

## 3 Relation Extraction Patterns

Both these approaches used extraction patterns which were based on dependency analysis (Tesniére, 1959) of text. Under this approach the structure of a sentence is represented by a set of directed binary links between a word (the head) and one of its modifiers. These links may be labelled to indicate the grammatical relation between the head and modifier (e.g. subject, object). Cyclical paths are generally disallowed and the analysis forms a tree structure. An example dependency analysis for the sentence "*Acme Inc. hired Mr Smith as their new CEO, replacing Mr Bloggs.*" is shown in Figure 1.

The extraction patterns used by both Yangarber et al. (2000) and Stevenson and Greenwood (2005) were based on SVO tuples extracted from dependency trees. The dependency tree shown in Figure 1 would generate two patterns: $\texttt{replace} \xrightarrow{obj} \texttt{Mr Bloggs}$ and $\texttt{Acme Inc.} \xleftarrow{subj} \texttt{hire} \xrightarrow{obj} \texttt{Mr Smith}$. While these represent some of the core information in this sentence, they cannot be used to identify a number of relations including the connection between *Mr. Smith* and *CEO* or between *Mr. Smith* and *Mr. Bloggs*.

A number of alternative approaches to constructing extraction patterns from dependency trees have been proposed (e.g. (Sudo et al., 2003; Bunescu and Mooney, 2005)). Previous analysis (Stevenson and Greenwood, 2006a) suggests that the most useful of these is one based on pairs of linked chains from the dependency tree. A chain can be defined as a path between a verb node and any other node in the dependency tree passing through zero or more intermediate nodes (Sudo et al., 2001). The linked chains model (Greenwood et al., 2005) represents extraction patterns as a pair of chains which share the same verb but no direct descendants. It can be shown that linked chain patterns can represent the majority of relations within a dependency analysis (Stevenson and Greenwood, 2006a). For example, the dependency tree shown in Figure 1 contains four named entities (*Acme Inc.*, *Mr Smith*, *CEO* and *Mr. Bloggs*) and linked chains patterns can be used to represent the relation between any pair.[1] Some example patterns extracted from the analysis in Figure 1 can be seen in Figure 2. An additional advantage of linked chain patterns is that they do not cause an unwieldy number of candidate patterns to be generated unlike some other approaches for representing extraction patterns, such as the one proposed by Sudo et al. (2003) where any subtree of the dependency tree can act as a potential pattern.

When used within IE systems these patterns are generalised by replacing terms which refer to specific entities with a general semantic class. For example, the pattern $\texttt{Acme Inc.} \xleftarrow{subj} \texttt{hire} \xrightarrow{obj} \texttt{Mr Smith}$ would become $\texttt{COMPANY} \xleftarrow{subj} \texttt{hire} \xrightarrow{obj} \texttt{PERSON}$.

## 4 Pattern Similarity

Patterns such as linked chains have not been used by semi-supervised approaches to pattern learning. These algorithms require a method of determining the similarity of patterns. Simple patterns, such as SVO tuples, have a fixed structure containing few items and tend to occur relatively frequently in corpora. However, more complex patterns, such as linked chains, have a less fixed structure and occur less frequently. Consequently, the previously proposed approaches for determining pattern similarity (see Section 2) are unlikely to be as successful with these more complex patterns. The approach proposed by Stevenson and

[1]Note that we allow a linked chain pattern to represent the relation between two items when they are on the same chain, such as *Mr Smith* and *CEO* in this example.

Greenwood (2005) relies on representing patterns as vectors which is appropriate for SVO tuples but not when patterns may include significant portions of the dependency tree. Yangarber et al. (2000) suggested a method where patterns were compared based on their distribution across documents in a corpus. However, since more complex patterns are more specific they occur with fewer corpus instances which is likely to hamper this type of approach.

Another approach to relation extraction is to use supervised learning algorithms, although they require more training data than semi-supervised approaches. In particular various approaches (Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005) have used kernel methods to determine the sentences in a corpus which contain instances of a particular relation. Kernel methods (Vapnik, 1998) allow the representation of large and complicated feature spaces and are therefore suitable when the instances are complex extraction rules, such as linked chains. Several previous kernels used for relation extraction have been based on trees and include methods based on shallow parse trees (Zelenko et al., 2003), dependency trees (Culotta and Sorensen, 2004) and part of a dependency tree which represents the shortest path between the items being related (Bunescu and Mooney, 2005). Kernels methods rely on a similarity function between pairs of instances (the kernel) and these can be used within semi-supervised approaches to pattern learning such as those outlined in Section 2.

## 4.1 Structural Similarity Measure

The remainder of this Section describes a similarity function for pairs of linked chains, based on the tree kernel proposed by Culotta and Sorensen (2004). The measure compares patterns by following their structure from the root nodes through the patterns until they diverge too far to be considered similar.

Each node in an extraction pattern has three features associated with it: the word, the relation to a parent, and the part-of-speech (POS) tag. The values of these features for node $n$ are denoted by $n_{word}$, $n_{reln}$ and $n_{pos}$ respectively. Pairs of nodes can be compared by examining the values of these features and also by determining the semantic similarity of the words. A set of four functions, $F = \{word, relation, pos, semantic\}$, is used to compare nodes. The first three of these correspond to the node features with the same name; the relevant function returns 1 if the value of the feature is equal for the two nodes and 0 otherwise. For example, the $pos$ function compares the values of the part of speech feature for nodes $n_1$ and $n_2$:

$$pos\,(n_1, n_2) = \begin{cases} 1 & \text{if } n_1, pos = n_2, pos \\ 0 & otherwise \end{cases}$$

The remaining function, $semantic$, returns a value between 0 and 1 to signify the semantic similarity of lexical items contained in the word feature of each node. This similarity is computed using the WordNet (Fellbaum, 1998) similarity function introduced by Lin (1998) .

The similarity of two nodes is zero if their part of speech tags are different and, otherwise, is simply the sum of the scores provided by the four functions which form the set $F$. This is represented by the function $s$:

$$s\,(n_1, n_2) = \begin{cases} 0 \text{ if } pos(n_1, n_2) = 0 \\ \sum\limits_{f \in F} f\,(n_1, n_2) \text{ otherwise} \end{cases}$$

The similarity of a pair of linked chain patterns, $l_1$ and $l_2$, is determined by the function $sim$:

$$sim\,(l_1, l_2) = \begin{cases} 0 \text{ if } s\,(r_1, r_2) = 0 \\ s\,(r_1, r_2) + \\ sim_c\,(C_{r_1}, C_{r_2}) \text{ otherwise} \end{cases}$$

where $r_1$ and $r_2$ are the root nodes of patterns $l_1$ and $l_2$ (respectively) and $C_r$ is the set of children of node $r$.

The final part of the similarity function calculates the similarity between the child nodes of $n_1$ and $n_2$.[2]

$$sim_c\,(C_{n_1}, C_{n_2}) = \sum_{c_1 \in C_{n_1}} \sum_{c_2 \in C_{n_2}} sim\,(c_1, c_2)$$

Using this similarity function a pair of identical nodes have a similarity score of four. Consequently, the similarity score for a pair of linked chain patterns can be normalised by dividing the similarity score by 4 times the size (in nodes) of the larger pattern. This results in a similarity function that is not biased towards either small or large patterns but will select the most similar pattern to those already accepted as representative of the domain.

This similarity function resembles the one introduced by Culotta and Sorensen (2004) but also

[2] In linked chain patterns the only nodes with multiple children are the root nodes so, in all but the first application, this formula can be simplified to $sim_c\,(C_{n_1}, C_{n_2}) = sim(c_1, c_2)$.

differs in a number of ways. Both functions make use of WordNet to compare tree nodes. Culotta and Sorensen (2004) consider whether one node is the hypernym of the other while the approach introduced here makes use of existing techniques to measure semantic similarity. The similarity function introduced by Culotta and Sorensen (2004) compares subsequences of child nodes which is not required for our measure since it is concerned only with linked chain extraction patterns.

## 5 Experiments

This structural similarity metric was implemented within the general framework for semi-supervised pattern learning presented in Section 2. At each iteration the candidate patterns are compared against the set of currently accepted patterns and ranked according to the average similarity with the set of similar accepted patterns. The four highest scoring patterns are considered for acceptance but a pattern is only accepted if its score is within 0.95 of the similarity of the highest scoring pattern.

We conducted experiments which compared the proposed pattern similarity metric with the vector space approach used by Stevenson and Greenwood (2005) (see Section 2). That approach was originally developed for simple extraction patterns consisting of subject-verb-object tuples but was extended for extraction patterns in the linked chain format by Greenwood et al. (2005). We use the measure developed by Lin (1998) to provide information about lexical similarity. This is the same measure which is used within the structural similarity metric (Section 4).

Three different configurations of the iterative learning algorithm were compared. (1) **Cosine (SVO)** This approach uses the SVO model for extraction patterns and the cosine similarity metric to compare them (see Section 2). This version of the algorithm acts as a baseline which represents previously reported approaches (Stevenson and Greenwood, 2005; Stevenson and Greenwood, 2006b). (2) **Cosine (Linked chain)** uses extraction patterns based on the linked chain model along with the cosine similarity to compare them and is intended to determine the benefit which is gained from using the more expressive patterns. (3) **Structural (Linked chain)** also uses linked chain extraction patterns but compares them using the similarity measure introduced in Section 4.1.

| |
|---|
| $\texttt{COMPANY}\xleftarrow{subj}\texttt{appoint}\xrightarrow{obj}\texttt{PERSON}$ |
| $\texttt{COMPANY}\xleftarrow{subj}\texttt{elect}\xrightarrow{obj}\texttt{PERSON}$ |
| $\texttt{COMPANY}\xleftarrow{subj}\texttt{promote}\xrightarrow{obj}\texttt{PERSON}$ |
| $\texttt{COMPANY}\xleftarrow{subj}\texttt{name}\xrightarrow{obj}\texttt{PERSON}$ |
| $\texttt{PERSON}\xleftarrow{subj}\texttt{resign}$ |
| $\texttt{PERSON}\xleftarrow{subj}\texttt{depart}$ |
| $\texttt{PERSON}\xleftarrow{subj}\texttt{quit}$ |

Table 1: Seed patterns used by the learning algorithm

### 5.1 IE Scenario

Experiments were carried out on the management succession extraction task used for the Sixth Message Understanding Conference (MUC-6) (MUC, 1995). This IE scenario concerns the movement of executives between positions and companies. We used a version of the evaluation data which was produced by Soderland (1999) in which each event was converted into a set of binary asymmetric relations. The corpus contains four types of relation: `Person-Person`, `Person-Post`, `Person-Organisation`, and `Post-Organisation`. At each iteration of the algorithm the related items identified by the current set of learned patterns are extracted from the text and compared against the set of related items which are known to be correct. The systems are evaluated using the widely used precision (P) and recall (R) metrics which are combined using the F-measure (F).

The texts used for these experiments have been previously annotated with named entities. MINIPAR (Lin, 1999), after being adapted to handle the named entity tags, was used to produce the dependency analysis from which the pattersn were generated. All experiments used the seed patterns in Table 1 which are indicative of this extraction task and have been used in previous experiments into semi-supervised IE pattern acquisition (Stevenson and Greenwood, 2005; Yangarber et al., 2000).

The majority of previous semi-supervised approaches to IE have been evaluated over preliminary tasks such as the identification of event participants (Sudo et al., 2003) or sentence filtering (Stevenson and Greenwood, 2005). These may be a useful preliminary tasks but it is not clear to what extent the success of such systems will be repeated when used to perform relation extraction. Conse-

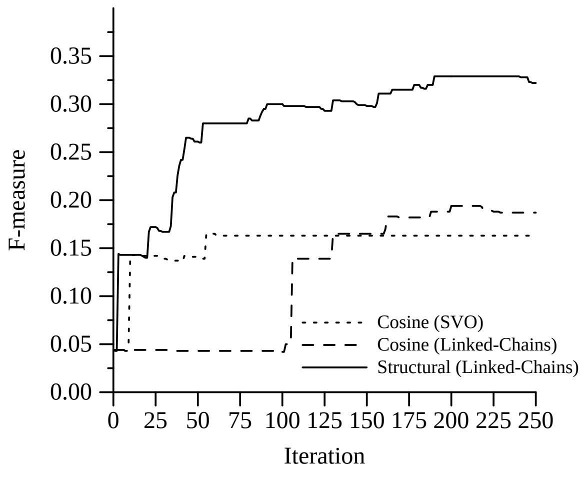

Figure 3: F-measure scores for relation extraction over 250 iterations

quently we chose a relation extraction task to evaluate the work presented here.

## 6 Results

Results from the relation extraction evaluation can be seen in Table 2 and Figure 3. The seven seed patterns achieve a precision of 0.833 and recall of 0.022. The two approaches based on cosine similarity performs poorly, irrespective of the pattern model being used. The maximum increase in F-measure of 0.15 (when using the cosine measure with the linked chain model) results in a maximum F-measure for the cosine similarity model of 0.194 (with a precision of 0.491 and recall of 0.121) after 200 iterations.

The best result is recorded when the linked chain model is used with the similarity measure introduced in Section 4.1, achieving a maximum F-measure of 0.329 (with a precision of 0.434 and recall of 0.265) after 190 iterations. This is not a high F-measure when compared against supervised IE systems, however it should be remembered that this represents an increase of 0.285 in F-measure over the original seven seed patterns and that this is achieved with a semi-supervised algorithm.

## 7 Conclusions

A number of conclusions can be drawn from the work described in this paper. Firstly, semi-supervised approaches to IE pattern acquisition benefit from the use of more expressive extraction pattern models since it has been shown that the performance of the linked chain model on the relation extraction task is superior to the simpler SVO model. We have previously presented a theoretical analysis (Stevenson and Greenwood, 2006a) which suggested that the linked chain model was a more suitable format for IE patterns than the SVO model but these experiments are, to our knowledge, the first to show that applying this model improves learning performance. Secondly, these experiments demonstrate that similarity measures inspired by kernel functions developed for use in supervised learning algorithms can be applied to semi-supervised approaches. This suggests that future work in this area should consider applying other similarity functions, including kernel methods, developed for supervised learning algorithms to the task of semi-supervised IE pattern acquisition. Finally, we demonstrated that this similarity measure outperforms a previously proposed approach which was based on cosine similarity and a vector space representation of patterns (Stevenson and Greenwood, 2005).

## Acknowledgements

This work was carried out as part of the RESuLT project funded by the Engineering and Physical Sciences Research Council (GR/T06391) and partially funded by the IST 6th Framework project X-Media (FP6-26978).

## References

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, B.C.

Aron Culotta and Jeffery Sorensen. 2004. Dependency Tree Kernels for Relation Extraction. In *42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database and some of its Applications*. MIT Press, Cambridge, MA.

Mark A. Greenwood, Mark Stevenson, Yikun Guo, Henk Harkema, and Angus Roberts. 2005. Automatically Acquiring a Linguistically Motivated Genic Interaction Extraction System. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, Bonn, Germany.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the Fif-*

| Iteration | Cosine (SVO) | | | Cosine (Linked Chains) | | | Structural (Linked Chains) | | |
|---|---|---|---|---|---|---|---|---|---|
| # | P | R | F | P | R | F | P | R | F |
| 0 | 0.833 | 0.022 | 0.044 | 0.833 | 0.022 | 0.044 | 0.833 | 0.022 | 0.044 |
| 25 | 0.600 | 0.081 | 0.142 | 0.833 | 0.022 | 0.044 | 0.511 | 0.103 | 0.172 |
| 50 | 0.380 | 0.085 | 0.139 | 0.500 | 0.022 | 0.043 | 0.482 | 0.179 | 0.261 |
| 75 | 0.383 | 0.103 | 0.163 | 0.417 | 0.022 | 0.043 | 0.484 | 0.197 | 0.280 |
| 100 | 0.383 | 0.103 | 0.163 | 0.385 | 0.022 | 0.042 | 0.471 | 0.220 | 0.300 |
| 125 | 0.383 | 0.103 | 0.163 | 0.500 | 0.081 | 0.139 | 0.441 | 0.220 | 0.293 |
| 150 | 0.383 | 0.103 | 0.163 | 0.500 | 0.099 | 0.165 | 0.429 | 0.229 | 0.298 |
| 175 | 0.383 | 0.103 | 0.163 | 0.481 | 0.112 | 0.182 | 0.437 | 0.247 | 0.315 |
| 200 | 0.383 | 0.103 | 0.163 | 0.491 | 0.121 | 0.194 | 0.434 | 0.265 | 0.329 |
| 225 | 0.383 | 0.103 | 0.163 | 0.415 | 0.121 | 0.188 | 0.434 | 0.265 | 0.329 |
| 250 | 0.383 | 0.103 | 0.163 | 0.409 | 0.121 | 0.187 | 0.413 | 0.265 | 0.322 |

Table 2: Comparison of the different similarity functions when used to perform relation extraction

*teenth International Conference on Machine learning (ICML-98)*, Madison, Wisconsin.

Dekang Lin. 1999. MINIPAR: A Minimalist Parser. In *Maryland Linguistics Colloquium*, University of Maryland, College Park.

MUC. 1995. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Mateo, CA. Morgan Kaufmann.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, pages 1044–1049, Portland, OR.

Stephen Soderland. 1999. Learning Information Extraction Rules for Semi-structured and free text. *Machine Learning*, 31(1-3):233–272.

Mark Stevenson and Mark A. Greenwood. 2005. A Semantic Approach to IE Pattern Induction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 379–386, Ann Arbor, MI.

Mark Stevenson and Mark A. Greenwood. 2006a. Comparing Information Extraction Pattern Models. In *Proceedings of the Information Extraction Beyond The Document Workshop (COLING/ACL 2006)*, Sydney, Australia.

Mark Stevenson and Mark A. Greenwood. 2006b. Learning Information Extraction Patterns using WordNet. In *Third International Global WordNet Conference (GWC-2006)*, Jeju Island, Korea.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2001. Automatic Pattern Acquisition for Japanese Information Extraction. In *Proceedings of the Human Language Technology Conference (HLT2001)*.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2003. An Improved Extraction Pattern Representation Model for Automatic IE Pattern Acquisition. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 224–231, Sapporo, Japan.

Lucien Tesniére. 1959. *Eleménts de Syntaxe Structurale*. Klincksiek, Paris.

Vladimir Vapnik. 1998. *Statistical Learning Theory*. John Wiley and Sons.

Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. 2000. Automatic Acquisition of Domain Knowledge for Information Extraction. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, pages 940–946, Saarbrücken, Germany.

Roman Yangarber. 2003. Counter-training in the Discovery of Semantic Patterns. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 343–350, Sapporo, Japan.

Dimitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.