# Automatically Acquiring a Linguistically Motivated Genic Interaction Extraction System

**Mark A. Greenwood** M.GREENWOOD@DCS.SHEF.AC.UK
**Mark Stevenson** M.STEVENSON@DCS.SHEF.AC.UK
**Yikun Guo** G.YIKUN@DCS.SHEF.AC.UK
**Henk Harkema** H.HARKEMA@DCS.SHEF.AC.UK
**Angus Roberts** A.ROBERTS@DCS.SHEF.AC.UK
Department of Computer Science, University of Sheffield, Sheffield, S1 4DP, UK

## Abstract

This paper describes an Information Extraction (IE) system to identify genic interactions in text. The approach relies on the automatic acquisition of patterns which can be used to identify these interactions. Performance is evaluated on the Learning Language in Logic (LLL-05) workshop challenge task.

## 1. Extraction Patterns

The approach presented here uses extraction patterns based on paths in dependency trees (Lin, 1999). Dependency trees represent sentences using dependency relationships linking each word in the sentence with the words which modify it. For example in the noun phrase *brown dog* the two words are linked by an adjective relationship with the noun *dog* being modified by the adjective *brown*. Each word may have several modifiers but each word may modify at most one other word.

In these experiments the extraction patterns consist of linked chains, an extension of the chain model proposed by Sudo et al. (2003) which represents patterns as any chain-shaped path in a dependency tree starting from a verb node. Our model extends this to patterns produced by joining pairs of chains which share a common verb root but no direct descendants. For example the fragment *"...*AGENT *represses the transcription of* TARGET*..."* can be represented by the dependency tree in Figure 1. From such a tree we extract all the chains and linked chains that contain at least one semantic category giving the 4 patterns (2 chains and 2 linked chains) shown in Table 1.

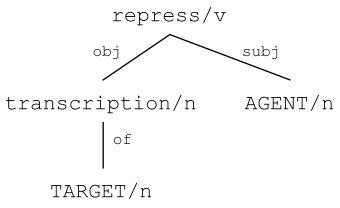


*Figure 1.* An example dependency tree.

The nodes in the dependency trees from which our patterns are derived can be either a lexical item or a semantic category such as gene, protein, agent, target, etc. Lexical items are represented in lower case and semantic categories are capitalised, e.g. in `verb[v/transcribe](subj[n/GENE]+obj[n/PROTEIN])`[1], `transcribe` is a lexical item while `GENE` and `PROTEIN` are semantic categories which could match any lexical item of that type. These patterns can be used to extract interactions from parsed text by matching against dependency trees.

## 2. Extraction Pattern Learning

Our approach learns patterns automatically by identifying those with similar meanings to a set of seed patterns known to be relevant. The motivation behind this approach is that language is often used to express the same information in alternative ways. For example *"*AGENT *represses the transcription of* TARGET*"*, *"the transcription of* TARGET *is repressed by* AGENT*"*, and *"*TARGET *(repressed by* AGENT*)"* describe the same interaction. Our approach aims to identify various ways interactions can be expressed by identifying patterns

Appearing in *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, Bonn, Germany, 2005. 

[1] In this pattern representation `+` signifies that two nodes are siblings and a nodes descendants are grouped within `(` and `)` directly after the node.

```
verb[v/repress](subj[n/AGENT])
verb[v/repress](obj[n/transcription](of[n/TARGET]))
verb[v/repress](obj[n/transcription]+subj[n/AGENT])
verb[v/repress](obj[n/transcription](of[n/TARGET])+subj[n/AGENT])
```

*Table 1.* The patterns extracted from the dependency tree in Figure 1.

which paraphrase one another. A similar method is outlined in more detail in Stevenson and Greenwood (2005).

Extraction patterns are learned using a weakly supervised bootstrapping method, similar to that presented by Yangarber (2003), which acquires patterns from a corpus based upon their similarity to patterns which are known to be useful. The general process of the learning algorithm is as follows:

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 although the algorithm has no access to this information.
2. This corpus is pre-processed to generate the set of all patterns which could be used to represent sentences contained in the corpus, call this set $S$. The aim of the learning process is to identify the subset of $S$ representing patterns which are relevant to the IE scenario.
3. The user provides a small set of seed patterns, $S_{seed}$, which are relevant to the scenario. These patterns are used to form the set of currently accepted patterns, $S_{acc}$, so $S_{acc} \leftarrow S_{seed}$. The remaining patterns are treated as candidates for inclusion in the accepted set, these form the set $S_{cand} (= S - S_{acc})$.
4. A function, $f$, is used to assign a score to each pattern in $S_{cand}$ based on those which are currently in $S_{acc}$. This function assigns a real number to candidate patterns so $\forall\ c \in S_{cand},\ f(c, S_{acc}) \mapsto \mathbb{R}$. 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 $S_{learn}$.
5. The patterns in $S_{learn}$ are added to $S_{acc}$ and removed from $S_{cand}$, so $S_{acc} \leftarrow S_{acc} \cup S_{learn}$ and $S_{cand} \leftarrow S_{acc} - S_{learn}$.
6. If a suitable set of patterns has been learned then stop, otherwise return to step 4.

The most important stage in this process is step 4; the task of identifying the most suitable pattern from the set of candidates. We do this by finding patterns that are similar to those already known to be useful. Similarity is measured using a vector space model inspired by that commonly used in Information Retrieval (Salton & McGill, 1983). Each pattern is represented as a set of pattern element-filler pairs. For instance, the pattern `verb[v/transcribe](subj[n/GENE]+obj[n/PROTEIN])` contains the pairs `verb_transcribe`, `subj_GENE` and `obj_PROTEIN`. The set of element-filler pairs in a corpus can be used to form the basis for a vector space in which each pattern can be represented as a binary vector (where the value 1 for a particular element denotes the pattern contains the pair and 0 that it does not). The similarity of two pattern vectors can be compared using Equation 1.

$$similarity(\vec{a}, \vec{b}) = \frac{\vec{a} W \vec{b^T}}{|\vec{a}||\vec{b}|} \tag{1}$$

Here $\vec{a}$ and $\vec{b}$ are pattern vectors, $\vec{b^T}$ the transpose of $\vec{b}$, and $W$ a matrix listing the semantic similarity between each of the possible pattern element-filler pairs which is crucial for this measure. Assume that the set of patterns, $P$, consists of $n$ element-filler pairs denoted by $p_1, p_2, ...p_n$. Each row and column of $W$ represents one of these pairs. So, for any $i$ such that $1 \leq i \leq n$, row $i$ and column $i$ are both labelled with pair $p_i$. $w_{ij}$ is the element of $W$ in row $i$ and column $j$ and is the similarity between $p_i$ and $p_j$. Pairs with different pattern elements (i.e. grammatical roles) have a similarity score of 0. The remaining elements of $W$ represent the similarity between the filler of pairs of the same element type. Similarity is determined using a metric defined by Banerjee and Pedersen (2002) which uses the WordNet lexical database (Fellbaum, 1998)[2]. This metric measures the relatedness of a pair of words by examining the number of words that are common in their definitions.

Figure 2 shows an example using three potential extraction patterns:

[2]This measure was chosen since it allows relatedness scores to be computed for a wider range of grammatical categories than alternative measures.

Extraction Patterns

a. `verb[v/block](subj[n/protein])`
b. `verb[v/repress](subj[n/enzyme])`
c. `verb[v/promote](subj[n/protein])`

Matrix Labels

1. `subj_protein`
2. `subj_enzyme`
3. `verb_block`
4. `verb_repress`
5. `verb_promote`

Similarity Matrix

| | | | | |
|---|---|---|---|---|
| 1 | 0.95 | 0 | 0 | 0 |
| 0.95 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0.9 | 0.1 |
| 0 | 0 | 0.9 | 1 | 0.1 |
| 0 | 0 | 0.1 | 0.1 | 1 |

Similarity Values

$\text{sim}(\vec{a}, \vec{b}) = 0.925$
$\text{sim}(\vec{a}, \vec{c}) = 0.55$
$\text{sim}(\vec{b}, \vec{c}) = 0.525$

*Figure 2.* Similarity scores and matrix for an example vector space using three patterns.

`verb[v/block](subj[n/protein])`
`verb[v/repress](subj[n/enzyme])`
`verb[v/promote](subj[n/protein])`

This example shows how these patterns can be represented as vectors and gives a sample semantic similarity matrix. It can be seen that the first pair of patterns are the most similar using the proposed measure despite the fact they have no lexical items in common.

The measure shown in Equation 1 is similar to the cosine metric, commonly used to determine the similarity of documents in the vector space model approach to Information Retrieval. However, the cosine metric will not perform well for our application since it does not take into account the similarity between elements of a vector and would assign equal similarity to each pair of patterns in this example[3].

The second part of a pattern element-filler pair can be a semantic category, such as `GENE`. The identifiers used to denote these categories do not appear in WordNet and so it is not possible to directly compare their similarity with other lexical items. To avoid this problem such tokens are manually mapped onto the most appropriate node in the WordNet hierarchy which is then used in similarity calculations.

An associated problem is that WordNet is a domain independent resource and may list several inappropriate meanings for domain specific words. For example WordNet lists five senses of the word *transcribe*, only one of which is related to the biomedical domain. To alleviate this problem domain specific restrictions are applied to WordNet. In these experiments only specific senses of 58 words are used with the alternative senses for each word being ignored by the system. These 58 words include the 30 verbs detailed in the PASBio project[4] (Wattarujeekrit et al., 2004) and 28 words determined by manual analysis of MedLine abstracts. For example, *transcribe* contains five senses in WordNet but our system considers only the final one; *convert the genetic information in (a strand of DNA) into a strand of RNA, especially messenger RNA*.

We experimented with several techniques for ranking candidate patterns to decide which patterns to learn at each iteration of our algorithm and found the best results were obtained when each candidate pattern was compared against the centroid vector of the currently accepted patterns. At each iteration we accept the four highest scoring patterns whose score is within 0.95 of the best pattern being accepted. For further details of the same approach using predicate-argument structures to perform sentence filtering, see Stevenson and Greenwood (2005).

## 3. Pattern Acquisition

Two training corpora were used for the experiments reported in this paper:

**Basic** The basic data set, without coreference, as provided by the LLL-05 challenge organizers.

**Expanded** The basic data set expanded with 78 automatically acquired *weakly labelled* (Craven & Kumlien, 1999) MedLine sentences. This extra training data was obtained by extracting, from MedLine abstracts[5] containing the phrase *Bacillus subtilis*, those sentences which contain two dictionary entries (or their synonyms) which are known to form an interaction in the basic training data.

The training corpora are pre-processed to produce one sentence per known interaction, replacing the agent and target by representative tags, `AGENT` and `TARGET`, and all other dictionary elements by the tag `OTHER`. The resulting sentences are then parsed using MINI-

[3]The cosine metric for a pair of vectors is given by the calculation $\frac{a.b}{|a||b|}$. Substituting the matrix multiplication in the numerator of Equation 1 for the dot product of vectors $\vec{a}$ and $\vec{b}$ would give the cosine metric. Note that taking the dot product of a pair of vectors is equivalent to multiplying by the identity matrix, i.e. $\vec{a}.\vec{b} = \vec{a}I\vec{b^T}$. Under our interpretation of the similarity matrix, $W$, this equates to saying that all pattern element-filler pairs are identical to each other and not similar to anything else.

[4]`http://research.nii.ac.jp/~collier/projects/PASBio/`

[5]Only abstracts which appeared after the year 2000 were used in order to comply with the LLL challenge guidelines.

PAR (Lin, 1999) to produce dependency trees from which the candidate extraction patterns (in the form of chains and linked chains) are extracted.

The learning algorithm was used to learn two sets of extraction patterns using the pair of corpora and the seed patterns in Table 2 which where chosen following a manual inspection of the training data. Due to the small amount of training data the learning algorithm was allowed to run until it was unable to learn any more patterns. When trained using the basic corpora the algorithm ran for 74 iterations and acquired 127 patterns. When trained using expanded corpora the algorithm ran for 130 iterations and acquired 236 patterns.

Not all the extraction patterns acquired in this way encode a complete interaction, i.e. they do not contain both `AGENT` and `TARGET` slots. To generate full interactions those agents and targets which are extracted are joined together using the following heuristics:

- Each `AGENT` extracted is paired with all the `TARGET` instances extracted from the same sentence (vice-versa for `TARGETS`).
- Each `AGENT/TARGET` discovered by a pattern is paired with the closest (distance measured in words) dictionary element.

For example imagine a sentence in which all the agents and targets discovered by extraction patterns are tagged as `AGENT` or `TARGET`, all other dictionary elements are replaced by `OTHER`: *`TARGET`*$_1$ *blocks `AGENT` and `OTHER` which inhibits `TARGET`*$_2$. From this sentence the following interactions would be extracted `AGENT`→`TARGET`$_1$, `AGENT`→`TARGET`$_2$ and `AGENT`→`OTHER`, i.e. the `AGENT` would be paired with all `TARGET` instances as well as the closest dictionary element.

## 4. A Baseline System

A baseline system was developed for comparison with our main approach. This baseline system assumes that interactions exist between all possible pairs of named entities in any given sentence (participants were provided with an exhaustive named entity dictionary). For instance, given a sentence containing three named entities labelled A, B and C, six interactions AB, AC, BA, BC, CA and CB are generated. This baseline will identify many interactions although the precision is likely to be low as many incorrect interactions will also be generated.

## 5. Evaluation

The official evaluation results, for both the baseline system and the systems trained using the two corpora detailed in Section 3, can be seen in Table 3.

We may expect the baseline system to achieve 100% recall by proposing a link between each pair of entities in each sentence. However certain constructions describe two relations between a pair of entities. For example *"...A activates or represses B..."* describes both repression and activation relationships between A and B while the baseline would propose just one.

In comparison with the baseline system our machine learning approach to pattern acquisition performed poorly due to low recall, although with a precision score over twice that of the baseline. The performance can probably be attributed to the small amount of available training data. It is clear that adding just a small amount of additional training data (78 sentences from MedLine) had a positive effect increasing the overall F-measure from 14.8% to 17.5%. The same effect can be seen if we consider the performance of the systems over the three interaction types; action, bind and regulon. The system trained using just the basic data finds 6 correct interactions 5 of which are actions and 1 a binding interaction (see Table 4 for a full breakdown of the results for all three submissions). The system fails to find any regulon family interactions. This is understandable given the training data which contains different percentages of each of the three interaction types. For instance only three sentences containing a regulon family interaction are provided illustrating just six interactions. Given our method of pattern acquisition this means that even if all the relevant patterns from these three sentences are learnt they would only apply to very similar sentences when used for extraction as they will not have been able to generalise far enough away from the specific instances present in the three example sentences.

### 5.1. Additional Evaluation

We carried out additional evaluations after the official results for the challenge task had been released.

A more detailed evaluation of the learning algorithm considers the performance of the patterns acquired at each separate iteration as opposed to the results in the previous section which evaluate all the acquired patterns as a single set. Figure 3 shows the F-measure score of the system trained using the expanded corpus (see Section 3) at each iteration of the learning algorithm.

This evaluation highlights a number of interesting

```
verb[v/transcribe](by[n/AGENT]+obj[n/TARGET])
verb[v/be](of[n/AGENT]+s[n/expression](of[n/TARGET]))
verb[v/inhibit](obj[n/activity](nn[n/TARGET])+subj[n/AGENT])
verb[v/bind](mod[r/specifically](to[n/TARGET])+subj[n/AGENT])
verb[v/block](obj[n/capacity](of[n/TARGET])+subj[n/AGENT])
verb[v/regulate](obj[n/expression](nn[n/TARGET])+subj[n/AGENT])
verb[v/require](obj[n/AGENT]+subj[n/gene](nn[n/TARGET]))
verb[v/repress](obj[n/transcription](of[n/TARGET])+subj[n/AGENT])
```

*Table 2.* Seed patterns used for pattern acquisition.

| System | P | R | F |
|---|---|---|---|
| Baseline | 10.6% (53/500) | 98.1% (53/54) | 19.1% |
| LLL-05 Basic | 22.2% (6/27) | 11.1% (6/54) | 14.8% |
| LLL-05 Expanded | 21.6% (8/37) | 14.8% (8/54) | 17.5% |

*Table 3.* Evaluation results of our three submissions.

| | All Interactions | | | Action | | | Bind | | | Regulon | | | No Interaction | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| System | C | M | S | C | M | S | C | M | S | C | M | S | C | M | S |
| Baseline | 53 | 1 | 447 | 35 | 1 | 95 | 14 | 0 | 46 | 4 | 0 | 6 | 0 | 0 | 300 |
| LLL-05 Basic | 6 | 48 | 21 | 5 | 31 | 7 | 1 | 13 | 2 | 0 | 4 | 0 | 0 | 0 | 12 |
| LLL-05 Expanded | 8 | 46 | 29 | 7 | 29 | 11 | 1 | 13 | 2 | 0 | 4 | 0 | 0 | 0 | 16 |

*Table 4.* Breakdown of the official evaluation results including results for individual interaction types (columns represent **C**orrect, **M**issing, and **S**purious). Precision = **C**/(**C**+**S**), Recall = **C**/(**C**+**M**)

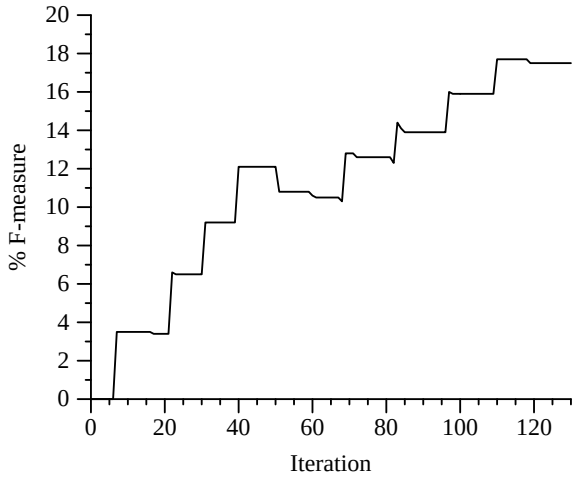


*Figure 3.* Increasing F-measure scores.

points. Firstly the seed patterns (Table 2) while being possibly representative of the training data do not match any of the interactions in the test set (i.e. the F-measure at iteration zero is 0% reflecting the fact that no correct interactions were extracted by the seed patterns). This is unfortunate as the learning algorithm is designed to acquire patterns which are similar in meaning to a set of known *good* patterns. In this instance, however, the algorithm started by acquiring patterns which are similar to the seeds but which clearly do not represent the interactions in the test set. However, this also means that those interactions extracted by the completed system were done so using only patterns acquired during training and not hand-picked good quality seed patterns.

The per-iteration evaluation in Figure 3 also shows that the learning algorithm is relatively stable even when inappropriate patterns are acquired. At least one pattern is acquired at each iteration and these results show that even if patterns are not able to extract valid interactions they rarely affect the performance of the current set of acquired patterns. The notable exception to this is at iteration 51 when a pattern is acquired which drops the F-measure from 12.1% to 10.8%, although further analysis shows that this was in fact a problem with the extraction procedure and not the acquired pattern. The algorithm acquired the pattern `verb[v/contain](obj[n/TARGET]+subj[n/AGENT])`. Un-

fortunately while the TARGET usually matches against a dictionary element the AGENT often matches other text. This causes the nearest (in words) dictionary element to be used as the AGENT which, in turn, can lead to incorrect interactions being extracted from text.

This analysis of the system's failings highlights a useful feature of our approach. Many machine learning algorithms produce classifiers which are statistical in nature and do not consist of a set of rules but rather a complex combination of probabilities. This makes it difficult to analyse classification mistakes and does not allow the ability to modify the classifier by removing badly performing rules. In contrast to this our approach learns human readable extraction rules which can be easily inspected, modified or removed to suit a given scenario. This allows an expert to examine the extraction rules while automating the time consuming process of rule acquisition.

### 5.2. Sentence Filtering

Our approach to automatically acquiring IE patterns has been shown to be suitable for determining the relevance of sentences for an extraction task in the management succession domain (Stevenson & Greenwood, 2005). The sentence filtering task involves using the set of acquired patterns to classify each sentence in a corpus as either relevant (containing the description of an interaction) or not. Sentence filtering is an important preliminary stage to full relation extraction. Using the patterns acquired from the expanded corpus (described in Section 3) we can also perform sentence filtering of the LLL challenge test data[6]. The results of this filtering, at different iterations of the algorithm, can be seen in Figure 4.

These results show that set of acquired patterns achieves an F-measure score of 47.5% resulting from precision and recall scores of 57.6% and 40.4% respectively. This compares to results reported by Nédellec et al. (2001) who achieve an F-measure score of approximately 80% over similar data using a supervised approach in which the learning algorithm was aware of the classification of the training instances. It should be noted that our approach was trained using only a small amount of unlabelled training data (181 sentences compared with approximately 900 sentences used by Nédellec et al. (2001)) and the sentence filtering results should be considered in this context.

[6]Thanks to Claire Nédellec for providing the relevant/not-relevant labelling of the sentences required for this evaluation.

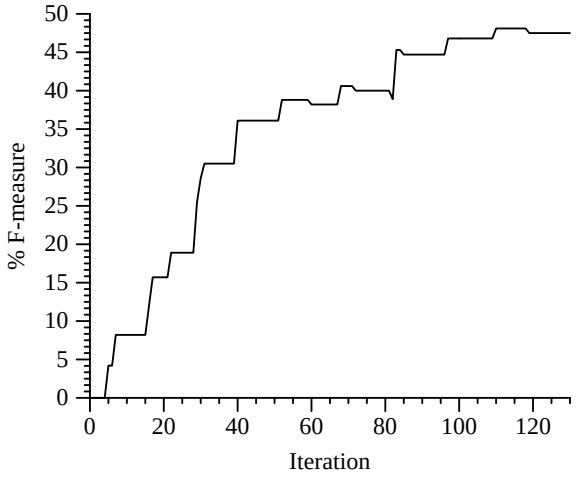


*Figure 4.* BioMedical Sentence Filtering.

## 6. Failure Analysis

The experiments reported in this paper have shown that our system is disappointing when used to perform relation extraction. The main failure of the system to extract meaningful relations can be traced back to the lack of training data. When extra data obtained from MedLine was also used to train the system there was an improvement in performance, acquiring more data may further improve performance. Another possible solution to this problem would be to generalise the acquired patterns in some form, perhaps by allowing any synonym of a pattern element filler to match. These could be extracted from WordNet.

One further source of failure was due to errors in the dependency trees introduced by MINIPAR. This is probably because the parser was not trained on biomedical texts and hence suffers from problems with unknown words and grammatical constructions. The approach here relies heavily on access to accurate dependency tree representations of text.

## 7. Conclusions

In this paper we have presented a linguistically motivated approach to extracting genic interactions from biomedical text. Whilst the performance of the system was disappointing achieving an F-measure score of only 17.5% we believe that the approach is well motivated but suffers from a lack of training data and parsing problems. We showed that increasing the training data using weakly labelled text did in fact increase the performance of the system. The additional evaluation of the extraction patterns showed that the approach is also resilient to the algorithm learning inappropriate extraction patterns.

## Acknowledgements

This work was carried out as part of the RESuLT project funded by the Engineering and Physical Sciences Research Council (GR/T06391).

## References

Banerjee, S., & Pedersen, T. (2002). An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. *Proceedings of the Fourth International Conference on Computational Linguistics and Intelligent Text Processing (CICLING-02)* (pp. 136–145). Mexico City.

Craven, M., & Kumlien, J. (1999). Constructing Biological Knowledge Bases by Extracting Information from Text Sources. *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology* (pp. 77–86). Heidelberg, Germany: AAAI Press.

Fellbaum, C. (Ed.). (1998). *Wordnet: An electronic lexical database and some of its applications*. Cambridge, MA: MIT Press.

Lin, D. (1999). MINIPAR: a minimalist parser. *Maryland Linguistics Colloquium*. University of Maryland, College Park.

Nédellec, C., Vetah, M. O. A., & Bessières, P. (2001). Sentence Filtering for Information Extraction in Genomics, a Classification Problem. *Proceedings of the Conference on Practical Knowledge Discovery in Databases (PKDD'2001)* (pp. 326–338). Freiburg, Germany.

Salton, G., & McGill, M. (1983). *Introduction to modern information retrieval*. New York: McGraw-Hill.

Stevenson, M., & Greenwood, M. A. (2005). A Semantic Approach to IE Pattern Induction. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*.

Sudo, K., Sekine, S., & Grishman, R. (2003). An Improved Extraction Pattern Representation Model for Automatic IE Pattern Acquisition. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)* (pp. 224–231).

Wattarujeekrit, T., Shah, P., & Collier, N. (2004). PASBio: Predicate-Argument Structures for Event Extraction in Molecular Biology. *BMC BioInformatics*, *5:155*.

Yangarber, R. (2003). Counter-training in the discovery of semantic patterns. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)* (pp. 343–350). Sapporo, Japan.