Detecting protein-protein interactions in biomedical texts using a parser and linguistic resources

Schneider, Gerold ; Kaljurand, Kaarel ; Rinaldi, Fabio

DOI: https://doi.org/10.1007/978-3-642-00382-0_33

Detecting Protein-Protein Interactions in Biomedical Texts using a Parser and Linguistic Resources

Gerold Schneider, Kaarel Kaljurand, and Fabio Rinaldi

Institute of Computational Linguistics, University of Zurich, Switzerland
gschneid@ifi.uzh.ch, kalju@ifi.uzh.ch, rinaldi@ifi.uzh.ch

Abstract. We describe the task of automatically detecting interactions between proteins in biomedical literature. We use a syntactic parser, a corpus annotated for proteins, and manual decisions as training material. After automatically parsing the GENIA corpus, which is manually annotated for proteins, all syntactic paths between proteins are extracted. These syntactic paths are manually disambiguated between meaningful paths and irrelevant paths. Meaningful paths are paths that express an interaction between the syntactically connected proteins, irrelevant paths are paths that do not convey any interaction.
The resource created by these manual decisions is used in two ways. First, words that appear frequently inside a meaningful paths are learnt using simple machine learning. Second, these resources are applied to the task of automatically detecting interactions between proteins in biomedical literature. We use the IntAct corpus as an application corpus.
After detecting proteins in the IntAct texts, we automatically parse them and classify the syntactic paths between them using the meaningful paths from the resource created on GENIA and addressing sparse data problems by shortening the paths based on the words frequently appearing inside the meaningful paths, so-called transparent words.
We conduct an evaluation showing that we achieve acceptable recall and good precision, and we discuss the importance of transparent words for the task.

1 Introduction

Scientific articles reporting results of biomedical studies are growing exponentially in number[1]. Publically available literature services such as Pubmed (http://pubmed.gov) already contain more than 17 million articles. Even for the expert it has become difficult to keep an overview of new results. Fully or partly automated systems that extract biological knowledge from text have thus become a necessity. Particularly, knowledge about protein-protein interactions

[1] This research is partially funded by the Swiss National Science Foundation (grant 100014-118396/1). Additional support is provided by Novartis Pharma AG, NITAS, Text Mining Services, CH-4002, Basel, Switzerland.

(PPI) is needed in biomedical and genetic research, as exemplified by the LLL genic interaction challenge [1] and the BioCreAtIvE challenge PPI track [2].

A number of methods have been applied to this task. Simple approaches classify two proteins as interacting when mentioned in the same sentence, or when their cooccurrence in an abstract is very frequent [3]. Such approaches often yield high recall at low precision and can be used as baselines for more involved approaches.

Other approaches apply handcrafted rules, for example regular expressions for surface searches [4], or syntactic patterns on automatically parsed corpora [5, 6]. These approaches typically achieve high precision at the cost of recall.

Third, machine learning methods are increasingly used to construct a model from large annotated sources. To extract meaningful features for the model construction, dependency parsing is often used. [7] extract sentences in which two proteins and an interaction word co-occur. Their features include the interaction words and the parents of the proteins, according to the dependency analysis. [8] use a walk kernel which contains fragments of the paths between two proteins. Due to sparse data, the paths were partitioned into patterns, each consisting of two vertices and their intermediate edge (*vertex-walk*), and of two edges and their common vertex (*edge-walk*). While this alleviates the sparse data problem, it neglects that many semantic configurations are not local, they depend on considerably larger tree fragments. We suggest to use a single feature consisting of the entire path, but to reduce reduce sparseness by using very little lexical information and linguistic insights to shorten the paths.

[9] extends the approach of [8] by using a feature-based approach instead of a kernel, where e.g. each vertex-walk and each edge-walk is a feature, on the one hand a lexical feature containing words, on the other hand a syntactic feature containing tags. The lexical features are quite sparse due to Zipf's law.

The approach that we present in this paper is hybrid. It uses a large, partly annotated resource and manual annotations. It uses parsed data in order to obtain a suitable level of abstraction and reducing the number of manual annotation decisions, thus creating a new linguistic resource. It achieves higher precision than coocurrence methods because it uses stricter requirements. It achieves higher recall than handcrafted syntactic patterns because all syntactic connections that are observed in a large corpus are taken into consideration. Machine Learning methods and backoff techniques are applied to the linguistic resource thus created.

For training, we have used the GENIA corpus, to which we have manually added interaction information. Our approach shows a new application of the GENIA corpus. For the application phase, we use the IntAct corpus[10]. The IntAct corpus was devised to be used for the PPI task but has been underused so far.

The aim of our application is twofold. On the one hand, we use the IntAct data as a gold standard for evaluating the performance of our PPI algorithm, on the other hand we propose an algorithm that may help IntAct annotators by suggesting protein-protein interactions to them.

Our approach is also characterised by using linguistic insights and lightweight resources, allowing us to achieve good results despite using simple statistical methods and learning algorithms.

The paper is structured as follows. We summarise our term detection and grounding method in chapter 2. In chapter 3, we describe how we collect and annotate the syntactic data. In chapter 4, our application to the IntAct corpus is described. We give an evaluation in chapter 5 and conclude in chapter 6.

2 Term Detection and Grounding

Term detection and grounding is a necessary preliminary step to the detection of interaction. Our approach is described in detail in [11] and summarised here.

We compiled a term list of 1,685,126 terms based on the terms extracted from UniProtKB[2], NCBI[3], and PSI-MI[4], The term list contains the term name, the term ID, and the term type in each entry. In this list, 934,973 of the terms are multi-word units.

2.1 Automatic Term Detection

Using the described term list, we can annotate biomedical texts in a straightforward way. First, the sentences and tokens are detected in the input text. We use the LingPipe tokenizer and sentence splitter which have already been trained on biomedical corpora. The term detector matches the longest possible and non- overlapping sequences of tokens in each sentence, and in the case of success, assigns all the possible IDs (as found in the term list) to the annotated sequence. The annotator ignores certain common English function words (we use a list of about 50 stop words). Also, figure and table references are detected and ignored.

In order to account for possible orthographic differences between the terms in the term list and the token sequences in the text, a normalization step is included. We apply standard normalization rules, such as removing all characters that are neither alphanumeric nor space, normalising Greek letters and Roman numerals, convert to lower case, apply biomedical normalizations, such as removing the final 'p' if it follows a number, e.g. 'Pan1p' → 'Pan1'.

2.2 Automatic Term Grounding

A marked up term can be ambiguous. The most frequent reason is that the term can be assigned several IDs from a single type. This happens very often with UniProtKB terms and is e.g. due to the fact that the same protein occurs in many different species. Such protein names can be disambiguated in various

[2] `http://www.uniprot.org`
[3] `http://www.ncbi.nlm.nih.gov/Taxonomy/`
[4] `http://psidev.sourceforge.net/mi/psi- mi.obo`

ways. We have combined two methods: (1: 'IntAct' in table 1) remove all the IDs that do not reference a species ID specified in a given list of species IDs, in this case the species found in the IntAct data; (2: 'span' in table 1) remove all IDs that do not agree with the IDs of the other protein names in the same textual span (e.g. sentence) with respect to the species IDs.

The 'span' method is motivated by the fact that according to the IntAct database, interacting proteins are usually from the same species: less than 2% of the listed interactions have different interacting species. Assuming that proteins that are mentioned in close proximity often constitute a mention of interaction, we used a simple disambiguation method: for every protein mention, the disambiguator removes every UniProtKB ID that references a species that is not among the species referenced by the IDs of the neighbouring protein mentions (we use same sentence as neighbourhood).

The disambiguation result is not always a single ID, but often just a reduced set of IDs. Also, it can happen that none of the IDs matches a listed species. In this case all the IDs are removed.

Table 1. Evaluation of term detection and grounding on IntAct, measured against PubMed IDs. Two forms of disambiguation were applied: IntAct = species list from Intact data; span = species of neighbouring proteins must match

Diamb. method	Precision	Recall	F-Score
No disamb.	3%	73%	5%
IntAct	56%	73%	63%
span	3%	71%	6%
IntAct & span	57%	72%	64%

We evaluated the accuracy of our automatic protein name detection and grounding method on a corpus provided by the IntAct project[5]. This corpus contains a set of 6198 short textual snippets (of 1 to about 3 sentences), where each snippet is mapped to a PubMed identier (referring to the article the snippet originates from), and an IntAct interaction identifier (referring to the interaction that the snippet describes). In other words, each snippet is a textual evidence that has allowed the curator to record a new interaction in the IntAct knowledge base. By resolving an interaction ID, we can generate a set of IDs of interacting proteins and a set of species involved in the interaction, for the given snippet. Using the PubMed identifiers, we can generate the same information for each mentioned article. By comparing the sets of protein IDs reported by the IntAct corpus providers, and the sets of protein IDs proposed by our tool, we can calculate the precision and recall values, as given in table 1.

[5] `ftp://ftp.ebi.ac.uk/pub/databases/intact/current/various/data-mining/`

3 Collection and Annotation of Syntactic Data

3.1 Parsing and Tree Walks

The GENIA corpus has been manually annotated for biomedical terms and proteins. It consists of 2,000 abstracts, containing over 18,000 sentences. We parse the GENIA corpus with a state-of-the art dependency parser which has been adapted to and evaluated on the biomedical domain [12, 13].

Figure 1 shows the output of the parser for the sentence

Significant amounts of Tom40 were also coprecipitated by anti - Tom20 and anti - Tom22

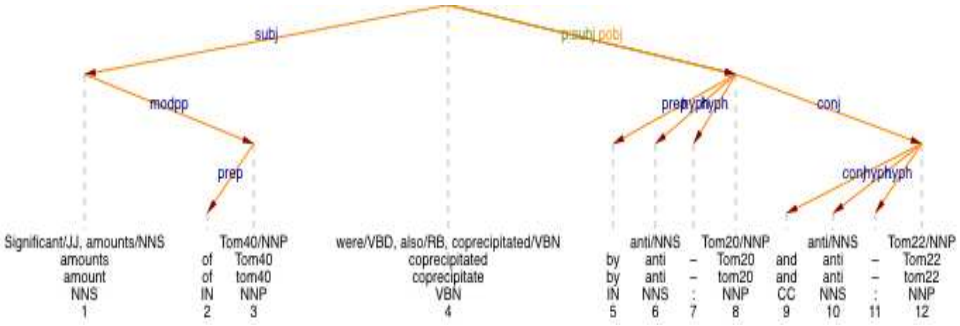

Fig. 1. Dependency parser output example

After parsing, we collect all syntactic connections that exist between all the terms as follows. For each term-coocurrence, i.e. two terms appearing in the same sentence, a collector traverses the tree from one term up to lowest common mother node, and down the second term, recording all intervening nodes. Such traversals have been used in many PPI applications [8], they are commonly called tree walks or paths. If one records all the information that an intermediate node contains, for example its lexical items and subnodes, the path would be extremely specific, which leads to sparse data and hence a recall problem for most applications. If one only records the grammatical role labels, the paths are too general, which leads to a precision problem for most applications. As a working assumption, we have recorded the lexical head lemma of the top node, and the grammatical labels plus prepositions connecting all intervening nodes. We have split the path into a left and a right half between the top node. The sentence in figure 1 contains 3 proteins: *Tom40*, *Tom20*, and *Tom22*. The path between *Tom40* and *Tom22* consists of the top node *coprecipitate*, the left path [*subj,modpp-of*] and the right path [*p:subj,conj*]. The path is treated as a single feature, unlike in most similar approaches, e.g. [8]. They use a a kernel with fragments of the of the paths between two proteins. Each pattern consists of two vertices and their intermediate edge (*vertex-walk*), and of two edges and

their common vertex (*edge-walk*). While this alleviates the sparse data problem, it neglects that many semantic configurations are not local, they depend on considerably larger tree fragments.

We suggest to use a single feature consisting of the entire path, but using very little lexical information and linguistic insights to shorten the paths. in [9], each vertex-walk and each edge-walk leads to two features, on the one hand a lexical feature containing words, on the other hand a syntactic feature containing tags. The lexical features are quite sparse due to Zipf's law. There is a small closed class of lexical items that is crucial to syntax [14, 15], namely prepositions, which we have thus introduced into the path. But also the syntactic features are potentially sparser than what is linguistically meaningful, as they contain tags. A subject relation, for example, is mostly between a noun and a verb. Since there are 4 noun tags and almost a dozen verb tags, sparseness is inflated. Our paths are also shorter and less sparse than in many other representations, because the syntactic graphs that we use are based on chunks. We present linguistic insights that further allow us to reduce data sparseness by shortening paths in section 4.2.

3.2 Manual Annotation

Only a minority of the paths extracted by the method just introduced actually express a biomedical interaction. The path between *Tom20* and *Tom22* in our example, which consists of the top node *Tom20*, the empty left node, and the right node [*conj*], does not express any biomedical interaction, it does not state in any way that *Tom20* and *Tom22* interact. In order to apply the paths to the PPI task we need to classify paths into those expressing an interaction relation and those that do not. We have decided to classify manually.

Ideally, one should classify every individual co-occurrence of two terms in the entire corpus. Since we did not have the resources to conduct such a large-scale, token-based annotation, we have opted for a type-based annotation at the level of the extracted paths. If our working assumption that these paths are a useful level of abstraction holds, the annotation task offers a useful compromise in the trade-off between token-wise annotation and unsupervised machine learning. We have discarded singletons, i.e. paths only appearing once in GENIA, since they are too sparse and often arise from parsing errors. The frequency-ranked list of paths tails off sharply, indicating a Zipfian distribution, more than half of all paths are singletons.

A major advantage of annotating a large corpus over formulating hand-written patterns is that no instance is missed (except for very rare ones that happen to be absent from a large corpus). This insight has given rise to the methodology of corpus linguistics in descriptive linguistics.

We manually annotated the about 2500 paths appearing at least twice. Each decision, i.e. whether the target path expresses a relation or rather not, was based on at least three example sentences[6] containing the target path. During

[6] except for paths that only appeared twice in GENIA

the annotation we observed that there are relatively few paths for which the example sentences suggested opposite decisions. We also observed that many paths express subset relations, for example *A is a B protein*, where *A* is a subset of *B*. We have decided to annotate these cases with a third class in addition to 'yes' and 'no', saving them for future ontology applications. Additionally, we observed that semantically lightweight nouns seem to play an important role for the decision: in order to test if a relation is expressed by an example sentence, it typically helps to paraphrase it, using the top node and the two terms. The sentence *A activates groups of B* essentially expresses that *A activates B*, or *A blocks activation of B*, or expresses that *A blocks B*, whereas *A activates C, which has a binding site for B* does not express that *A activates B*. There is a large set of words like *group* and *activation*, for which we would like to use the term *transparent words*. We complied lists of them and extended it with a simple machine-learning approach described in section 3.3.

We noticed that there are some paths, especially relatively long ones, for which it is very difficult to decide. When asked for the decision if *D3* interacts with *CD28*, based on the following example sentence, we found it hard to decide and agreed to opt for 'no' in difficult cases.

Further, engagement inhibited progression through the cycle by inhibiting the production of D3, kinase cdk, and cdk6 when the cells were stimulated with CD28 and with anti-CD3 alone.

3.3 Learning Transparent Words from the Type-Based Annotation

Although we collected the paths based only on the head lemma of the top node and the labels of intervening nodes, we also kept record of all intervening words in order to be able to learn specific rules where necessary. All the words intervening inside a path are, for instance, candidates for being transparent words, as introduced in section 3.2. For each word appearing inside a path, we calculate a score which simply divides its frequency inside a path by its total frequency. Words above a threshold are treated as transparent in the application phase.

4 Application to IntAct

The paths that are extracted from GENIA can directly be used for PPI detection. We chose the IntAct data as a gold standard. Although we have constructed the paths in a way that aims to reduce sparseness, and although we have used a corpus-based annotation method instead of introspective creation of hand-crafted rules, recall is very poor when the patterns are applied directly. The following are the main reasons why recall is low. In each subsection we also discuss how we have improved the situation.

4.1 Term Recognition and Upper Bound

The protein detectiontion and grounding algorithm which we use (section 2) has a recall of about 72%. Since any interaction involves two proteins, the per-

formance for recognition and grounding of protein pairs can be expected to be about 50% recall. It is beyond the scope of this paper to improve term recognition performance, so we need to accept this upper bound for the PPI task[7].

4.2 Transparent Words

Sparse data problems could be reduced significantly by applying the transparent words resource that we have created. If no annotated path from GENIA exists, the following sparse data reduction methods are used as backoffs:

- First, proteins occurring inside noun chunks are allowed to replace the head of the chunk if the head is an transparent word.
- Secondly (if still no path from GENIA exists), the relations for appositions, conjunctions and hyphens are cut.
- Third (if still no path from GENIA exists), parts of trees that are headed by an transparent word are cut.

In the example sentence 2 cutting conjunctions (second backoff) means that *portion of Tim54* appears at the same level as *Tim12*, cutting the transparent words *portion* and *all* (third backoff) means that *Tim54* appears at the same level as *Tim12*, and *Tim22* is only one PP-attachment (*modpp*) lower.

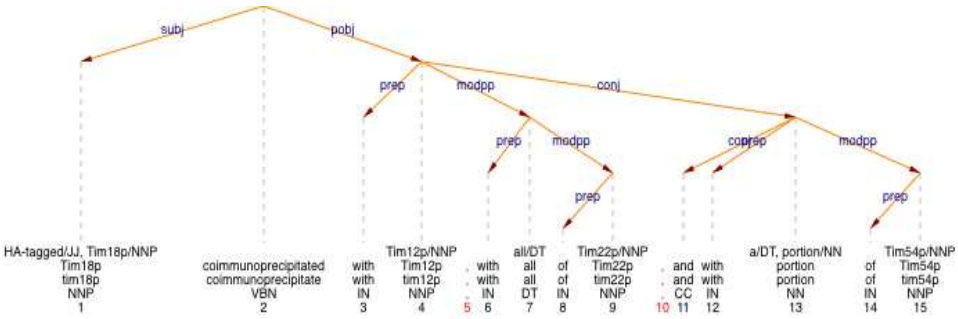

Fig. 2. Dependency parser output example

4.3 Surface Patterns to Address Tagging and Parsing Errors

Tagging and parsing errors are quite frequent, despite using taggers and parsers that are adapted to the domain. They are an important reason for the remaining sparseness. We have therefore developed surface patterns. They apply only at a backoff level, i.e. if no syntactic path is found in GENIA even after the syntactic backoffs.

There are three patterns. Each pattern consists of two proteins and a keyword:

[7] baseline 1 in the evaluation section calculates the exact upper bound

– A verb B, e.g *A interacts-with B*
– noun A B, e.g. *association between A and B*
– A B noun, e.g. *A - B binding*

The distance between *A* and the keyword, as well as the distance between *B* and the keyword are restricted, as is typical for observation window-based approaches. These surface patterns typically achieve relatively good precision, but insufficient recall. If observation windows are very large, recall increases but precision drops off.

5 Evaluation

We have evaluated our approach, as well as an upper bound and a number of baselines, in order to measure the relative success of our approach. We have used the first 1000 sentences of the IntAct data for the evaluation. We have mentioned that, given the performance of about our term recognition and grounding tool, the upper bound is about 50% recall. The term grounding tool sometimes delivers one UniProt ID and sometimes several UniProt IDs, on average 2.02 IDs. Since the ultimate aim of our approach is to deliver one and exactly one ID we speak of exact precision and recall if both proteins are given only one ID, and if there is an interaction, and of loose precision and recall if one or both of the proteins are given more than one ID by the grounding algorithm (i.e. if the UniProt ID could not be fully disambiguated), if one of the delivered IDs is correct for each protein, and there is an interaction. UniProt IDs are very fine-grained, including the organism in which the protein functions (ortholog). Since the task described in this paper is interaction detection rather than full term grounding disambiguation, we will mainly report loose precision and recall figures.

Our currently best system achieves 80.5% loose precision and 21.0% loose recall, and 59% exact precision at 15% exact recall. In order to assess the relative success that these performance figures mean, we will now compare them to a number of increasingly more advanced baselines.

5.1 Baselines

Baseline 1 Cooccurrence of two proteins in a sentence is a low baseline, one that heavily overgenerates. Precision of this baseline tells one how much one gets for free, while recall tells one how good one can maximally get (upper bound). We achieve 39.2% loose precision and 41.2% loose recall. Compared to this baseline, our best system has more than doubled precision at the cost of losing about half of the recall.

Baseline 2 In a purely syntactic approach, measuring all syntactically connected proteins lead to a second baseline. Since the parser we use does not always deliver analysis spanning the entire sentence, especially when sentences are complex, this is not a variant of baseline 1. We achieve 50.1% loose precision and 29.8% loose recall with baseline 2.

Baseline 3 In a purely "non-syntactic" surface based approach, observation windows are often used. We apply the surface patterns introduced in section 4.3, but no paths, and do not use information on transparent words. The window size is 5 words, which means that maximally 3 words may occur between the head (e.g. the verb) and the term. We achieve 78.6% loose precision and 11.4% loose recall with baseline 3.

Baseline 4 We extend baseline 3 by using the transparent words resource that we have created. This baseline is still purely surface-based and window size is 5, but transparent words are cut from the observation window, which means that remote words may move into the observation window if they are mainly separated by transparent words. We achieve 81.8% loose precision and 18.7% loose recall with baseline 4.

Best system The currently best system achieves 80.5% precision and 21.0% loose recall. It uses syntactic patterns, surface patterns and the transparent words resource at both levels.

Table 2. Baselines compared to the system

Method	**Description**	**Loose Precision**	**Loose Recall**
Baseline 1	sentence cooccurrence	39.2%	41.2%
Baseline 2	syntactically conncected	50.1%	29.8%
Baseline 3	surface, no transparent words	78.6%	11.4%
Baseline 4	surface, transparent words	81.8%	18.7%
Best system	syntax, surfaces, transparent words	80.5%	21.0%

The step-wise improvements from the baselines to the currently best system are summarised in table 2. The performance of the last baseline, surface-based but using transparent words, is impressive. Adding the transparent words resource to the system increased performance more than the syntactic filter which the best system uses. The best system achieves 51% of the upper bound recall in baseline 1. Given gold-standard term information, the system therefore, all other parameters being equal, achieves a performance of 80.5% precision and 51% recall on the PPI detection task, which amounts to an F-score of 62.4%.

5.2 Breakdown of Results

We have broken down the precision results in table 3, which also quantifies the backoff method we use. If a syntactic method gives a decision, it is used, otherwise same chunk is applied. If that does not give a decision, the surface patterns are used. When available, the syntax-based method delivers the highest precision,

Table 3. Breakdown of results

Backoff used	Loose Precision Percent	Loose Precision Count
Syntax	83.8%	62/74
Same chunk	75%	3/4
surface A verb B	76%	38/50
surface noun A B	82.4%	14/17
surface A B noun	66.7%	2/3
TOTAL	80.5%	120/149

but the surface method with the transparent words resource performs almost equally well. Absolute numbers are given in the third column.

We have tested a further syntactic backoff, which allows a path with a different top node to be used. Precision of this backoff was between 50 and 60%, lower than the surface backoff. We have also constrained the syntactic backoff using Wordnet, e.g. enforcing that the top node word for which a path was found in GENIA is similar to the candidate top node word, but precision did not increase.

6 Conclusions

We have created three new resources: annotated paths from the GENIA corpus, automatically learnt transparent words, and transparent words noted while annotating and testing. We have applied the resources to IntAct, both as a PPI task, and in order to develop an algorithm helping annotators.

We have evaluated our algorithm and performed better than all baselines. On the PPI task, our best system achieves 80.5% precision and 21% recall. 21% recall corresponds to 51% of the upper bound, if gold standard term recognition were used. We have based our path representations on linguistic insights. We use syntactic paths as features with very little lexical information (only the top node word and prepositions in PPs), and based on chunks, both of which lead to fewer sparse data problems. We have shown that transparent words, words with low semantic content, play an important role in allowing us to further reduce sparseness: we have cut transparent words and their nodes from our path representations. Further reducing sparseness by also excluding the top node word negatively affected performance.

References

1. Nedellec, C.: Learning language in logic – genic interaction extraction challenge. In: Proceedings of LLL '05. (2006) 31–37
2. Krallinger, M., Leitner, F., Rodriguez-Penagos, C., Valencia, A.: Overview of the protein-protein interaction annotation extraction task of biocreative ii. Genome Biology **9(Suppl 2)** (2008)

3. Rebholz-Schuhmann, D., H.Kirsch, Arregui, M., Gaudan, S., Riethoven, M., P.Stoehr: EBIMed – text crunching to gather facts for proteins from Medline. Bioinformatics **23(2)** (2006) e237 – e244
4. Giuliano, C., Lavelli, A., Romano, L.: Exploiting shallow linguistic information for relation extraction from biomedical literature. In: Proceedings of EACL 2006. (2006)
5. Rinaldi, F., Schneider, G., Kaljurand, K., Hess, M., Romacker, M.: An environment for relation mining over richly annotated corpora: the case of GENIA. BMC Bioinformatics **7(Suppl 3):S3** (2006)
6. Fundel, K., Küffner, R., Zimmer, R.: RelEx – relation extraction extraction using dependency parse trees. Bioinformatics **23(3)** (2007) 365–371
7. Erkan, G., Ozgur, A., Radev, D.R.: Extracting interacting protein pairs and evidence sentences by using dependency parsing and machine learning techniques. In: Proceedings of BioCreAtIvE 2. (2007)
8. Kim, S., Yoon, J., Yang, J.: Kernel approaches for genic interaction extraction. Bioinformatics **9:10** (2008)
9. Landeghem, S.V., Saeys, Y., de Peer, Y.V.: Extracting protein-protein interactions from text using rich feature vectors and feature selection. In: Proceedings of the Third International Symposium on Semantic Mining in Biomedicine (SMBM 2008), Turku, Finland (2008)
10. Kerrien, S., Alam-Faruque, Y., Aranda, B., Bancarz, I., Bridge, A., Derow, C., Dimmer, E., Feuermann, M., Friedrichsen, A., Huntley, R., Kohler, C., Khadake, J., Leroy, C., Liban, A., Lieftink, C., Montecchi-Palazzi, L., Orchard, S., Risse, J., Robbe, K., Roechert, B., Thorneycroft, D., Zhang, Y., Apweiler, R., Hermjakob, H.: Intact: open source resource for molecular interaction data. Nucleic Acids Res **(35 Database)** (2006) D561–D565
11. Kaljurand, K., Rinaldi, F., Kappeler, T., Schneider, G.: Detecting and grounding terms in biomedical literature. In: CICLing 2009, 10th International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, Mexico (2009)
12. Schneider, G., Kaljurand, K., Rinaldi, F., Kuhn, T.: Pro3Gres parser in the CoNLL domain adaptation shared task. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, Prague (2007) 1161–1165
13. Haverinen, K., Ginter, F., Pyysalo, S., Salakoski, T.: Accurate conversion of dependency parses: targeting the stanford scheme. In: Proceedings of Third International Symposium on Semantic Mining in Biomedicine (SMBM 2008), Turku, Finland (2008)
14. Collins, M., Brooks, J.: Prepositional attachment through a backed-off model. In: Proceedings of the Third Workshop on Very Large Corpora, Cambridge, MA (1995)
15. Collins, M.: Head-driven statistical models for natural language parsing. Computational Linguistics **29** (2003) 589 – 637