

---

# Learning Biological Interactions from Medline Abstracts

---

**Sophia Katrenko**

KATRENKO@SCIENCE.UVA.NL

Human-Computer Studies Laboratory, University of Amsterdam, Kruislaan 419, 1098 VA, Amsterdam

**M. Scott Marshall**

MARSHALL@SCIENCE.UVA.NL

Integrative Bioinformatics Unit, University of Amsterdam, Kruislaan 318, 1098 SM Amsterdam

**Marco Roos**

ROOS@SCIENCE.UVA.NL

Integrative Bioinformatics Unit, University of Amsterdam, Kruislaan 318, 1098 SM Amsterdam

**Pieter Adriaans**

PIETERA@SCIENCE.UVA.NL

Human-Computer Studies Laboratory, University of Amsterdam, Kruislaan 419, 1098 VA, Amsterdam

## Abstract

In this paper, we describe our approach to the Genic Interaction Extraction Challenge. Our solution combines several elements: 1) a domain theory about the interaction between language, semantics and syntax, 2) a biological ontology identifying amongst other things biomolecular entities and directed interaction verbs in the lexicon, 3) the notion of lexical-semantic-syntactic unification, 4) the notion of partial unification of lexical-semantic-syntactic trees and 5) the application of the standard RIPPER algorithm to the results. Using this approach on the very limited training and test data from the Challenge we show results that are promising. Our method observes a clear separation between domain-independent and domain-specific components. It can therefore easily be extended to other domains. We briefly describe the implementation of the techniques, discuss the results and give suggestions for improvements and further results in the conclusion.

## 1. Introduction

Extracting interactions from texts is an active field of research in the biomedical domain (Krallinger, M., 2005, for a recent review). Interactions between proteins and genes are often considered essential in the

description of biomolecular phenomena, and networks of (protein) interactions are considered as an entrée for a Systems Biology approach (Uetz, P. 2005). Interaction networks extracted from literature (Chen, H. 2004) complement interaction data obtained from high-throughput laboratory experiments (Xenarios, I., 2001). Our approach falls into a class that explores the application of semantic models (Yandell, M. D. 2002; Muller, H. M. 2004).

The paper is organized as follows. We first define the task and data supplied. Then, we proceed with the overview of the system which has been developed. The paper concludes with results received on the Challenge data and gives an outlook.

## 2. Challenge task

### 2.1. Biology

The Genic Interaction Extraction Challenge is aimed at learning interactions between agents and targets described by individual sentences that have been extracted from a Medline collection of abstracts. As is typical in articles about biology, the abstracts describe the results from various types of experiments and use a mix of biological viewpoints and jargon. The sentences have been extracted from different abstracts, thus there is no relation among the sentences on the discourse level.

For the purposes of the Challenge, a simplified notion of 'interaction' is used: an agent/target pair of (the names of) genes and/or proteins. Gene names can pertain to genetic entities such as the DNA code for a given gene or to physical entities, in which case they

may indirectly refer to gene products, i.e. proteins. However, it might happen that the same name for a biomolecular entity can refer to both the gene and the gene product and thereby perform the role of both agent and target. On the other hand, it is also possible to encounter more than one agent for a given target and vice versa.

A typical sentence with explicit interaction stated is the following:

*Localization of SpoIIE was shown to be dependent on the essential cell division protein FtsZ.*

where *SpoIIE* is the target and *FtsZ* is the agent and the interaction is represented as *genic\_interaction(FtsZ,SpoIIE)*. As one may notice, this relation is not symmetric.

We created a simple ontology specifically for use in this Challenge. The main purpose of the ontology was as a proof of principle for the use of semantics during text mining.

## 2.2. Data

There were 77 annotated training sentences provided by the organizers of the Challenge. Because we feel that syntactic information is vital, we only worked with the enriched training sets. An example sentence contains the following elements: a sentence-ID, the sentence itself, a corresponding list of words, a list of lemmas (stemmed versions of the words), a list of syntactic relations, a list of agents, targets and interactions. They can be grouped according to the presence/absence of the co-references into 22 sentences with co-references and 55 without them. The test set contains 87 sentences in the same format but without the agent-target information. In contrast to the training set, it also contains negative examples.

## 3. System overview

Following (Russell & Norvig 2003), the general knowledge-based inductive learning has to solve the following constraint:

$$\text{Background} \wedge \text{Hypothesis} \wedge \text{Descriptions} \\ \models \text{Classifications}$$

In our case, Background can be thought of as ontology and the knowledge it encapsulates, Descriptions are examples (and the data representation chosen). Hypothesis is unknown and has to be consistent with the background knowledge and observations.

The system we have designed includes several components as described further. The main idea of the sys-

tem has been inspired by the domain theory described in (Adriaans, 1992; Adriaans, 1991). This theory provides a framework for syntactic and semantic language learning under the assumption of compositionality. It allows the learning algorithm to deal with partial information on various levels of language description.

### 3.1. Data representation

In the context of the LLL-Challenge, learning is achieved by operations on so-called lss-structures:

```
lss_structure := lss(WordId,Word,Lemma,
SemanticType,SyntacticType)
```

In the rest of this paper we will use pseudo-Prolog notation to illustrate the basic concepts. A fundamental operation on lss-structures is lss-unification:

```
lss_univ(lss(_,Word,Lemma,Sem,Syn),
lss(_,Word,Lemma,Sem,Syn),1):-
\+Word = unknown,!.
lss_univ(lss(_,_,Lemma,Sem,Syn),
lss(_,_,Lemma,Sem,Syn),0.75):-
\+Lemma = unknown,!.
lss_univ(lss(_,_,_,Sem,Syn),
lss(_,_,_,Sem,Syn),0.5):-
\+ Sem = unknown,!.
lss_univ(lss(_,_,_,_,Syn),
lss(_,_,_,_,Syn),0.25):-
\+ Syn = unknown,!.
lss_univ(lss(_,_,_,_,_),lss(_,_,_,_,_),0):-!.
```

The lemmas and the syntactic types are derived from the annotation provided by the Challenge organizers. The semantic information was obtained from a limited domain-specific ontology that was developed in order to classify concepts encountered in the Challenge (OWL was imported to Prolog using a tool called Thea). Essential concepts such as "BioMolecularEntity" and "BiologicalProcess" were used as semantic tags for words. Classification of text instances such as those in a "named-entity dictionary" (provided by the organizers) was done by hand. A list of verbs was also classified as "DirectedActionVerb" and "InteractionVerb". In an effort to model the homomorphism between semantic and syntactic objects and thereby expose the semantics of syntactic operations, we included a small syntactic ontology in our system.

Lss-structures distinguish between levels of unification. The unification level is 1 when the words and all the other elements unify, 0.75 if the the words do not unify but the rest of the terms do unify, etc. The Prolog code above illustrates the idea. Along the same lines one can define partial unification be-

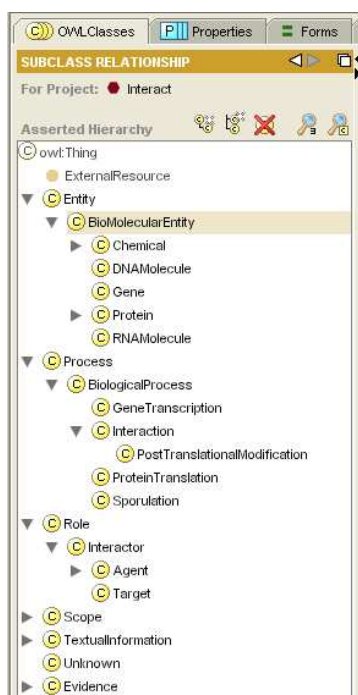


Figure 1. A high-level view of the mini-ontology.

tween two sets of lss-structures. For two sets  $A$  and  $B$ ,  $\text{lss\_univ}(A, B)$  is the maximal sum of lss-unifications between elements of unique pairs from  $A \times B$ , normalized for the cardinality of the biggest set. The value is 1 if  $A$  and  $B$  are the same and  $[0,1)$  otherwise. Syntactic dependency relations are represented in the form of lss-links:

```
lss_link(l1ss(_, Word1, Lemma1, Sem1, Syn1),
        SyntacticRelation,
        l1ss(_, Word2, Lemma2, Sem2, Syn2))
```

The definitions for lss-unification can easily be extended to unification for lss-link sets that define partial tree structures.

As input, the system receives sentences accompanied by their syntactic analysis in the form of dependency trees coded as lists of lss-links. The training examples have additional agent-target information. For the test examples this information has to be predicted. The learning process performed on the training set continues along the following lines:

1. For each pair of valid agent-target interactions  $\alpha$  and  $\beta$  in a sentence  $t$  in the training set we create a proto-rule  $\Gamma$  consisting of the lss-link set related to  $t$ , together with the information about the interaction between  $\alpha$  and  $\beta$ .

2. For each proto rule  $\Gamma$  predicting interaction between  $\alpha$  and  $\beta$ , and each combination of possible targets and agents,  $\gamma$  and  $\delta$ , in each sentence  $t$  of the training set, a *unification signature* is created. The unification signature is a list of unification values for specific parts of lss-link trees related to the location of  $\alpha$ ,  $\beta$ ,  $\gamma$  and  $\delta$  in the two trees. It also contains the score for the proto rule  $\Gamma$  on  $t$ : valid if the interaction between  $\gamma$  and  $\delta$  is the same as the one between  $\alpha$  and  $\beta$ , invalid otherwise.
3. A rule induction program (RIPPER) is applied to the unification signatures to identify useful combinations of proto rules. The resulting set of rules is a set of classifiers that can be applied to the test set.

A unification signature between two lss-link trees is a sequence of real values which reflect weights for the unification chosen by an expert. In case there is no unification, the system outputs 0. Based on the unification results, it is possible to apply different machine learning methods, in our case it is a rule induction.

Taken into account that the training set has not explicitly included negative examples, we have followed the closed world assumption. This way the cartesian product of all agents and targets has been computed over each sentence resulting in 873 combinations, 161 of which are positive examples. The proto-rules have been composed based on these positive examples from the training set. Cartesian product over potential agents and targets for the test set resulted in 568 combinations.

Unification which has been employed in this approach can be thought of in two dimensions. The first dimension presents unification to be carried out on a word and is defined on three levels, in particular, on the lexical, semantic and part of speech levels. While computing the scores we have taken into account the number of levels the unification proceeds on.

On the other hand, the unification can also take place on each level of the syntactic tree. In this case, tree levels are chosen by an expert.

Since it is nearly impossible to perform unification on the complete trees, it has been decided to consider partial trees. From this point of view, the most important parts are roots, ancestors for agent and target, bottom common ancestor for agent and target, and the parent and children for both:

```
rule(ID:A:T:Value,DUA,DDA,DUT,DDT,AncA,AncT,
     CAnC,BCAnC,RootA,RSynA,RootT,RSynT)
```

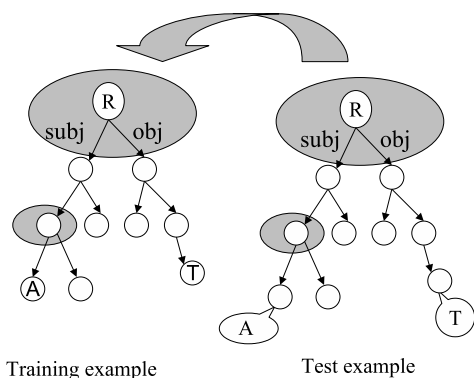


Figure 2. Unification on syntactic trees.

where  $ID, A, T, Value$  are sentence identifier, agent, target, and value (true if an interaction is present, false otherwise). In case of the test example,  $Value$  equals unknown.  $DUA, DDA, AncA$  ( $DUT, DDT, AncT$ ) are a parent, children and all ancestors for a given agent (target), whereas  $CAnc$  ( $BCanc$ ) is a list of common ancestors (bottom common ancestor). The values for  $RootA, RootT, RSynA, RSynT$  reflect the unification on the root level (the choice of two roots, for an agent and a target has been made assuming the existence of partial syntactic analysis).

For instance, the unification depicted on Fig.2 succeeds on the root level and on the parent level for an agent. Note that it has not been shown how well it succeeds on each of them.

### 3.2. Classification of interactions

After the unification has been carried out, it is necessary to perform classification. Since we have used weights while performing unification, the result is a sequence of real numbers. Provided that each test example has been unified on all proto-rules, it becomes necessary to reduce the output to a single sequence which has been done by averaging and normalizing. Some of the sequences can be safely eliminated before applying any classification method. They include such unification results for a given test example which have the same scores for each classification level considering two combinations, agent-target and target-agent. Having the same scores for both directions presupposes the potential agent and target to be siblings, which would be unlikely if they were true agent and target.

In order to classify potential interactions into negative and positive interactions, the Weka (Witten & Frank, 2000) implementation of the Ripper rule learner was used. It allows us to construct propositional rules

Table 1. Classification results for Ripper classifier on training and test data sets.

DATA SET	PRECISION	RECALL	F-MEASURE
TRAINING			
POSITIVE	71	55.3	62.2
NEGATIVE	90.5	94.9	92.7
TEST			
POSITIVE	39.2	26.5	31.6

based on the result of unification. This learner has first been tested on the training set making use of 10fold cross validation. The results for both classes are presented in Table 1. After fixing a bug that was present at the time of official testing we found that our results improved from 51,8% (precision), 16,8% (recall) and 25,4% (F-score) (official score) to 39,2%, 26,5% and 31,6% respectively. Please note that the results for the test data have been achieved testing our approach using the full data set (i.e., including coreferences). The precision and recall for the positive and negative examples in the training set have been calculated by us whereas the precision and recall for the test data (only positive examples) have been obtained by using score computation program provided by the organizers. The way of computing scores for the training and test sets slightly differs since for the test data the concept of spurious combinations has been introduced. Spurious combinations are only the elements of negative combinations set and they weren't explicitly marked in the training set.

Using rule induction has several advantages, including the ability to test which levels of unification are the most important for recognizing agents and targets. For example, it has turned out that it is not necessary to use all ancestors for an agent and a target (adding this information usually increases noise), it is sufficient to consider bottom common ancestors. The recognized interactions are of different types, as indicated in Table 2. One of the advantages of our approach lies in the ability to find interactions in the sentences with coreferences. Learning such interactions is considerably more difficult task in comparison to the interactions in the sentences without co-reference or ellipsis.

The lss-unification proved to be useful when comparing to the learning from examples without unification. In this case, the F-score dropped to 42,1% on the training set (for the class of positive interactions). We have also tested the usefulness of using ontology as a background knowledge. As it has turned out, it helped us

Table 2. Distribution of the interaction types .

CO-REFERENCES	INTERACTION TYPES		
	ACTION	BINDING	REGULON
YES	8	2	1
NO	5	5	1
TOTAL	13	7	2

to overcome the data sparseness, allowing for the unification on the semantic level even though the actual words or lemmas could be different.

There have been several other learners tested on the Challenge data. In this case, we used the same representation as for the proto-rules (e.g., information about the parents, children, etc.). One of such ILP systems was Aleph (Muggleton, S. H., 1994). However, it didn't provide better results. While increasing precision, the recall dropped significantly since only few interactions (around 10, depending on the settings) have been found. Thus, it would be interesting to test other representation including more information on the path from the agent to the target.

#### 4. Discussion

In this paper, we present an approach that incorporates not only the underlying syntactic structure of the data but also accounts for its semantics. The use of an ontology for semantic annotation enables us to disclose domain knowledge that might be useful for rule induction as well as reason about the rules that have been induced. It also gives us the opportunity to employ unification based on the structure of the ontology, by setting restrictions on the level of unification using the knowledge from the ontology (e.g., unification of elements having the same parent in the hierarchy).

Our ontology was initially simple in order to test ideas within the limited scope of the Challenge. However, a more refined model of the biological domain would strengthen our approach. For instance, we could take into account that not all interactions are of the agent/target type, or that an interaction is sometimes with some aspect of a gene rather than with the gene itself (e.g. with 'localisation of gene X'). We could also model different viewpoints used by biologists, such as the physical (molecular) viewpoint and the genetic one (Demerec, M. 1966). In general, our approach allows one to study the effect of various ontology design decisions.

There are several additional issues to be studied in

more detail in the future. First, we would like to augment the current manual semantic tagging with more classification rules for domain-specific concepts, such as interaction verbs and biological processes (e.g. post-translational modification). The additional semantic classification rules could then be applied during a second pass of semantic tagging, in order to enrich the annotation that enables our rules to classify interaction. Additionally, we would like to capture interaction that is stated indirectly, a form of 'undisclosed knowledge' (Swanson, D. R. 1986). There are two types of indirect interaction that we would like to capture: transitive interaction and collective interaction. Presumably, transitive interaction will be captured by forward chaining on a knowledgebase of interactions. However, the indirect and nondeterministic nature of such interactions calls into question the exclusive use of first-order logic; perhaps a probabilistic network model could be useful for this purpose. Such a network model would fit into our plans for a more sophisticated classification mechanism, where the results from several learners could be used to determine confidence in interactions. The second type of indirect interaction, collective interaction, is interaction with an entire class of entities. For instance, if we know that Protein A affects sporulation in general and that Protein B and C are produced during sporulation, we could conclude that Protein A affects Protein B and C. A mechanism to represent collective interaction could be combined with biological annotation that has been mapped into our ontological terms in order to effectively carry out the logic contained in the above example. More specifically, we could add a sporulation property to each protein whose functional category annotation includes sporulation, where sporulation is defined in our ontology.

Another interesting issue is related to interactions that have qualifying phrases such as "... in the mother cell during late stage sporulation". In this case, the interaction is constrained to a certain location and during a certain stage of development. The constraints in this particular example call for a set of locations and developmental stages to be taken up in the ontology, as well as a way to represent the qualifying information in the semantic tagging system.

From a molecular biology point of view molecular interactions form the basis of many phenomena and are essential to their description. This makes interaction extraction a good starting point in the larger endeavor of knowledge capture from biological documents. Of course, we would eventually like to use interactions alongside other types of extracted knowledge to build and extend knowledge models and ontologies. Sabou et al (Sabou, M. 2005) provide an example of building

a specialized ontology from a small corpus of web service descriptions. A similar approach could be potentially useful in e-science, where there is often a general shortage of resources for the annotation of experimental data.

## Acknowledgment

This work was carried out in the context of the Virtual Laboratory for e-Science project ([www.vl-e.nl](http://www.vl-e.nl)). This project is supported by a BSIK grant from the Dutch Ministry of Education, Culture and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ).

## References

- Adriaans, P. W. (1992). *Language Learning from a Categorical Perspective*. Doctoral dissertation, Proefschrift.
- Adriaans, P. W. (1991). *A Domain Theory for Categorical Language Learning Algorithms*. In Proceedings of the Eighth Amsterdam Colloquium, Amsterdam, The Netherlands. Editors Dekker, Paul and Stokhof, Martin. Institute for Logic, Language and Computation, University of Amsterdam.
- Chen, H., & B. M. Sharp. (2004). *Content-rich biological network constructed by mining PubMed abstracts*. BMC Bioinformatics 5: 147.
- Demerec, M., E. A. Adelberg, A. J. Clark, & P. E. Hartman. (1996). *A proposal for a uniform nomenclature in bacterial genetics*. Genetics 54: 61-76.
- Krallinger, M., R. A. Erhardt, & A. Valencia. (2005). *Text-mining approaches in molecular biology and biomedicine*. Drug Discov Today 10: 439-445.
- Muggleton, S. H. & L. De Raedt. (1994). *Inductive Logic Programming: Theory and Methods*. Logic Programming Journal, 19-20: 629-679.
- Muller, H. M., E. E. Kenny, & P. W. Sternberg. (2004). *Textpresso: an ontology-based information retrieval and extraction system for biological literature*. PLoS Biol 2: e309.
- Russell, S., & Norvig, P. (2003). *Artificial Intelligence. A Modern Approach. Second Edition*. New Jersey: Prentice Hall.
- Sabou M., Wroe C., Goble C., & Mishne G. (2005). *Learning Domain Ontologies for Web Service Descriptions: an Experiment in Bioinformatics*. In Proceedings of the 14th International World Wide Web Conference (WWW2005), Chiba, Japan, 10-14 May, 2005.
- Swanson, D. R. (1986). *Fish oil, Raynaud's syndrome, and undiscovered public knowledge*. Perspect Biol Med 30: 7-18.
- Uetz, P., and R. L. Finley, Jr. (2005). *From protein networks to biological systems*. FEBS Lett 579: 1821-1827.
- Witten, I. H., & Frank, E. (2000). *Data Mining: Practical machine learning tools with Java implementations*. San Francisco: Morgan Kaufmann.
- Xenarios, I., and D. Eisenberg. (2001). *Protein interaction databases*. Current Opinion in Biotechnology 12: 334-339.
- Yandell, M. D., and W. H. Majoros. (2002). *Genomics and natural language processing*. Nat Rev Genet 3: 601-610.