Advances in Deep Parsing of Scholarly Paper Content

Ulrich Schäfer and Bernd Kiefer

Language Technology Lab
German Research Center for Artificial Intelligence (DFKI)
Campus D3 1, D-66123 Saarbrücken, Germany
{ulrich.schaefer,kiefer}@dfki.de
http://www.dfki.de/lt

Abstract. We report on advances in deep linguistic parsing of the full textual content of 8200 papers from the ACL Anthology, a collection of electronically available scientific papers in the fields of Computational Linguistics and Language Technology.
We describe how – by incorporating new techniques – we increase both speed and robustness of deep analysis, specifically on long sentences where deep parsing often failed in former approaches. With the current open source HPSG (Head-driven phrase structure grammar) for English (ERG), we obtain deep parses for more than 85% of the sentences in the 1.5 million sentences corpus, while the former approaches achieved only approx. 65% coverage.
The resulting sentence-wise semantic representations are used in the Scientist's Workbench, a platform demonstrating the use and benefit of natural language processing (NLP) to support scientists or other knowledge workers in fast and better access to digital document content. With the generated NLP annotations, we are able to implement important, novel applications such as robust semantic search, citation classification, and (in the future) question answering and definition exploration.

1 Introduction

Scientists in all disciplines are nowadays faced with a flood of new publications every day. In addition, more and more publications from the past become digitally available and thus even increase the amount of data. Therefore, finding relevant information and avoiding redundancy and duplication of work have become urgent issues to be addressed by the scientific community.

The organization and preservation of scientific knowledge in scientific publications, vulgo text documents, thwarts these efforts. From a viewpoint of a computer scientist, scientific papers are just 'unstructured information'.

Automatically precomputed, normalized semantic representations of textual utterances could help to structure the search space and find equivalent or related propositions even if they are expressed differently, e.g. in passive constructions, using synonyms etc. Domain-relevant semantic similarity can be computed automatically and exploited as additional knowledge source to support robust search.

* Pre-print. The original publication is available at http://www.springerlink.com.

To again constrain the so expanded search space, users can ask the system in simply structured subject-predicate-object queries and get all matching, precomputed predicate-argument structures along with the original sentence from the paper. On the other hand, by storing the structure along with the original text in a structured full-text search engine such as Apache Lucene, it can be guaranteed that recall cannot fall behind the baseline of a fulltext search engine.

The basis of our scientific paper corpus is a subset of the ACL Anthology[1], a collection of conference and workshop papers in the field of Computational Linguistics and Language Technology. We concentrate on 8200 papers from the years 2002 through 2009 from which we extracted the textual content using Abbyy PDF Transformer.

Except for named entity recognition which is partly based on instances and concepts of a domain ontology, the processing pipeline we describe below is independent of the science domain.

To make the deep parser robust, it is embedded in a hybrid NLP workflow starting with a tokenizer, a part-of-speech tagger, and a named entity recognizer. These components help to identify and classify open class words such as person names, events (e.g. conferences) or locations. The trigram-based tagger helps to guess part-of-speech tags of words unknown to the deep lexicon. For both unknown words and named entities, generic lexicon entries are generated in the deep parser running the open source broad-coverage grammar ERG [5].

In contrast to shallow parsers, the ERG not only handles detailed syntactic analyses of phrases, compounds, coordination, negation and other linguistic phenomena that are important for extracting semantic relations, but also generates a formal semantic representation of the meaning of the input sentence in the MRS (Minimal Recursion Semantics; [6]) representation format. Ambiguities resulting in multiple readings per input sentence are ranked using a statistical parse ranking model.

In an earlier experiment, we obtained full deep parses for 64.89% of 955,581 sentences and 35.11% of the sentences were parsed by a fall-back shallow parser. Only 0.24% of the sentences could not be parsed at all.

In this chapter, we describe the fine-grained mapping of punctuation and other tokenization details by means of a chart mapping technique [1] ensuring that this information is now optimally used by the deep grammar for disambiguation. We also report on progress that we achieved by applying a chart pruning technique [7] that, as already proven on another corpus, helps to considerably increase parsing speed of the deep parser and the number of successfully parsed sentences. With both techniques applied together, we could not only increase parsing speed considerably, but also the coverage on the ACL Anthology corpus to more than 85%.

This chapter is structured as follows. In section 2, we present the improved parsing approach and results. In Section 3, we describe the semantic search application based on the improved parsing results. Section 4 discusses related work, and we finally conclude and give an outlook to future work in Section 5.

[1] `http://www.aclweb.org/anthology`

2 Deep Parsing of Scholarly Papers

The general idea of the semantics-oriented access to scholarly paper content is to apply NLP analysis to each sentence they contain and distill a structured representation that can be searched for in addition to fulltext. Different levels of analysis such as part-of-speech (PoS) tagging, named entity recognition (NER), chunking, shallow and deep parsing are suitable for different tasks.

While citation sentence classification in scholarly papers, a further application described in [16], is currently based on shallow NLP tasks such as tokenization, PoS tagging and patterns thereof only, the semantic search application is based on the full range of hybrid, robustness-oriented NLP. This includes shallow preprocessing with statistical taggers up to full deep parsing with generation of sentence semantics representations from which basically predicate-argument structure is derived. Thus, both applications share the preprocessing, and in the future, also citation sentence classification could make use of linguistic features extracted by more advanced NLP.

2.1 The Corpus

The basis of our scientific paper corpus is a subset of the ACL Anthology [2], a collection of conference and workshop papers in the field of Computational Linguistics and Language Technology. We concentrate on 8200 papers from the years 2002 through 2009 available in a native PDF format, i.e. not optically scanned at limited quality such as many older papers. Except for named entity recognition which is partly based an a domain ontology, the processing pipeline we describe below is independent of the science domain. However, we expect improvements in the future by modeling domain knowledge, e.g. through automatically extracted domain specific terms and ontology concepts.

2.2 PDF Extraction

The preprocessing step starts extracting clean text from the digital PDF documents. In a first version, we used PDFBox[2] to gain raw text content from the papers. This works well for most (especially recent) papers. However, it is problematic in general because PDFBox relies on the logical, digital content of the page (layout) description language PDF. Its internal structure is very much dependent on the tool that was used to generate the PDF, and there are many tools and of varying quality. Thus, decoding text from it does not work 100% correctly, and imposes severe problems up to complete garbage because of non-standard character encodings or no output on about 10% of the corpus.

To overcome these problems and become independent of the PDF encoder that was used to generate the digital paper, we recently moved to OCR-based PDF extraction with the commercial product Abbyy PDF Transformer[3]. It also

[2] `http://pdfbox.apache.org`
[3] `http://www.abbyy.com`

reliably resolves hyphenated words using its own language model as well as text (order) in tables. Moreover, and in contrast to PDFBox, it also works on scanned documents, provided that the scan quality is good enough. However, recognition of non-Latin characters such as in mathematical formulae remains a problem. It can be ignored for the time being because the NLP tools used also do not understand mathematics.

After text extraction, a sentence splitter segments into sentence units in order to provide suitable input for subsequent NLP. For each sentence, we record a unique document ID (in case of our corpus the ACL Anthology paper ID, e.g. C02-1023 for a paper from the COLING-2002 proceedings), the page on which it appeared, and the sentence number relative to the whole document. Amongst others, this information is important to highlight a search result or citation sentence within the original PDF paper layout.

2.3 Hybrid Parsing

To make the deep parser robust, it is embedded in a hybrid NLP workflow implemented using the hybrid NLP platform Heart of Gold [15]. Heart of Gold is an XML-based middleware architecture for the integration of multilingual shallow and deep natural language processing components, developed under the umbrella of the DELPH-IN initiative[4].

The employed Heart of Gold configuration instance starts with a tokenizer, the shallow part-of-speech tagger TnT [3] and the named entity recognizer SProUT [8]. These components help to identify and classify open class words such as person names, events (e.g. conferences) or locations.

The (trigram-based) tagger helps to guess part-of-speech tags of words unknown to the deep lexicon. For both unknown words and named entities, generic lexicon entries are generated in the deep parser. By means of the PET input chart XML format FSC [1], the shallow preprocessing results are combined and passed to the high-speed HPSG [12] parser PET [4] running the open source broad-coverage grammar ERG [5] (cf. Fig 2).

2.4 Precise Preprocessing Integration with Chart Mapping

Chart mapping [1] is a novel mechanism for the non-monotonic, rule-based manipulation of chart items that are described by feature structures. There are currently two chart mapping phases in PET during parsing: (1) Token mapping, where input items as delivered by external preprocessors are adapted to the expectations of the grammar. This requires that input items are described by feature structures – the *token feature structures*. (2) Lexical filtering, where lexical items can be filtered by hard constraints after lexical parsing has finished.

Token mapping requires tokens to be described by feature structures. Token feature structures can be arbitrarily complex. This allows users to pass information of various preprocessing modules into the parser. To this end, a new format, the XML-based FSC input format, was developed.

[4] `http://www.delph-in.net/heartofgold/`

Following is an excerpt from the FSC for the sentence "Resnik and Smith (2003) extract bilingual sentences from the Web to create parallel corpora for machine translation." (from anthology document N07-1043) generated by Heart of Gold preprocessing from TnT and SProUT output.

```
<fsc version="1.0">
  <chart id="hog://session1284321397757/collection1/TnT">
    <lattice init="v0" final="v20">
      <edge source="v0" target="v1">
        <fs type="token">
          <f name="+FORM"><str>Resnik</str></f>
          <f name="+FROM"><str>0</str></f>
          <f name="+TO"><str>6</str></f>
          <f name="+TNT">
            <fs type="tnt">
              <f name="+TAGS" org="list"><str>NNP</str></f>
              <f name="+PRBS" org="list"><str>1.000000</str></f>
            </fs>
          </f>
        </fs>
      </edge>
      ... <!-- more token edges from TnT -->
      <edge source="v6" target="v7">
        <fs type="token">
          <f name="+FORM"><str>extract</str></f>
          <f name="+FROM"><str>24</str></f>
          <f name="+TO"><str>31</str></f>
          <f name="+TNT">
            <fs type="tnt">
              <f name="+TAGS" org="list"><str>VB</str></f>
              <f name="+PRBS" org="list"><str>1.000000</str></f>
            </fs>
          </f>
        </fs>
      </edge>
      ... <!-- more token edges from TnT -->
      <!-- this edge comes from the Named Entity Recognizer -->
      <edge source="v0" target="v6">
        <fs type="token">
          <f name="+FORM"><str>Resnik and Smith (2003)</str></f>
          <f name="+FROM"><str>0</str></f>
          <f name="+TO"><str>23</str></f>
          <f name="+TNT"><fs type="null_tnt"/></f>
          <f name="+CLASS"><fs type="proper_ne"/></f>
          <f name="+TRAIT"><fs type="generic_trait"/></f>
        </fs>
      </edge>
    </lattice>
  </chart>
</fsc>
```

Figure 1 shows how tokenized and PoS-tagged input is combined with possibly concurrent information from a named entity recognizer, in the example SProUT delivering hypothetical information on named entities (here a citation string) in a single named entity item spanning over multiple words.

Concerning punctuation, the deep grammar can e.g. make use of information on opening and closing quotation marks. This information is often not explicit in the input text, e.g. when gained through OCR techniques, which make no distinction between ‘ and ’ or “ and ”. However, a tokenizer can often guess (reconstruct) leftness and rightness correctly. This information, passed to the deep parser via FSC, helps it to disambiguate.

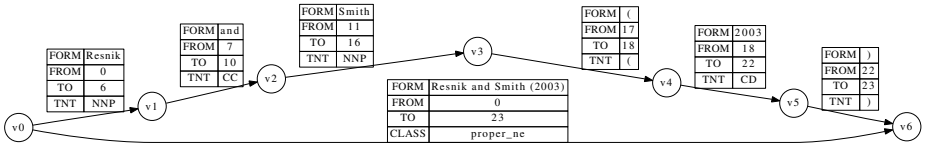


Fig. 1. FSC input to PET with combined information from tokenizer, PoS tagger and concurrent SProUT citation string item for input fragment “Resnik and Smith (2003) extract ...”

Furthermore, a new way of generic lexical instantiation has been introduced with token feature structures and chart mapping. In this new setup, the parser tries to instantiate all generic lexical entries for each word. Upon lexical instantiation, the token feature is unified into a designated path of the lexical entry. Only if this unification succeeds, the lexical item is instantiated. In order to control the instantiation of generic lexical entries, the token feature structures are appropriately constrained in the generic lexical entry, for instance by requiring that a generic verbal entry is only applicable for token feature structures where the highest ranked part-of-speech tag is a verb.

2.5 Increased Processing Speed and Coverage through Chart Pruning

The use of statistical models for result selection is well established for parsing with PET and ERG. We use a discriminative maximum entropy model based on WeScience data [9] to compute the best parse results. Recently, [7] described the use of a generative model to increase efficiency by shaping the search space of the parser towards the more likely constituents and pruning very unlikely ones. This method not only results in lower parse times, but also in slightly better coverage, since sentences which could not be parsed due to timeouts now fit into the given time bounds.

The generative model is in fact a probabilistic context-free grammar (PCFG) computed from the same tree banks as the discriminative model. The parser in PET is a straightforward bottom-up chart parser with agenda, which makes it

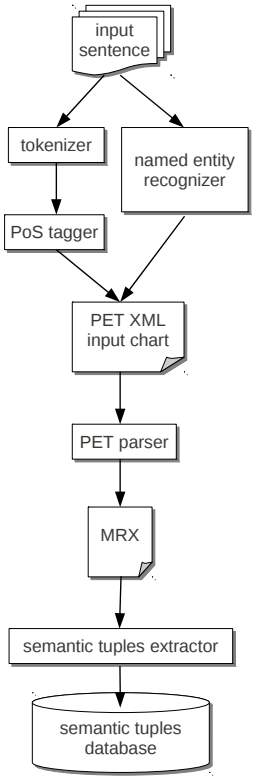


Fig. 2. Heart of Gold workflow for hybrid parsing and semantic tuples extraction

easy to use a model that has only local dependencies, such as PCFG. What is missing is a heuristics to prune unlikely items in a way that has a small computation overhead and will retain most of the items that are needed for the globally best results.

[11] did a very thorough comparison of different performance optimization strategies, and among those also a local pruning strategy which is similar to the one used by [7]. It restricts the number of items given both their length and start point in the chart. This is easy to implement and avoids the use of complicated heuristics to compensate the bias that shorter items become over longer chart items because of decreasing probability, which leads, without compensation, to a breadth-first strategy for the whole parse. The number of items per chart cell is restricted to a fixed number to hinder the parser from getting lost in local probability maxima.

There is an important difference to the system of [11], namely that their system works on a reduced context-free backbone of the grammar and then reconstructs the full results, while PET uses the full HPSG grammar directly,

with subsumption packing and partial unpacking to achieve a similar effect as the packed chart of a context-free parser.

The local chart pruning results in a measurable speed-up with a negligible decrease in parsing accuracy; in fact, an increase in f-measure has been observed because complicated sentences that had originally failed due to resource restrictions could now be parsed.

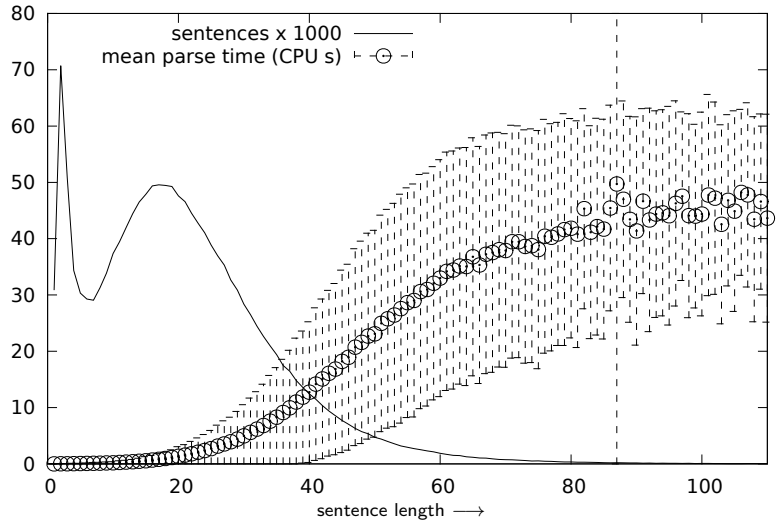


Fig. 3. Distribution of sentence length and mean parse times for mild pruning

Processing Results. In total, we parsed 1,537,801 sentences, of which 57,832 (3.8%) could not be parsed because of lexicon errors which are mostly due to OCR artifacts.

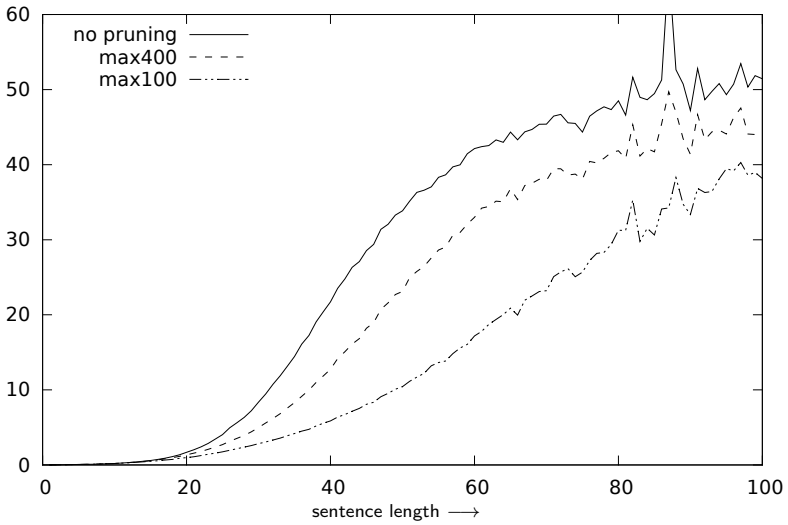
Figure 3 displays the average parse time of processing with moderate chart pruning, together with the mean quadratic error. In addition, it contains the distribution of input sentences over sentence length. Obviously, the vast majority of sentences has a length up to 60 words maximum.

Parse time was limited to 60 CPU seconds, and main memory consumption to 4 GB, which was far more than ever needed by the processes. Overall, the parse times only grow mildly due to the many optimization techniques in the original system, and also the new chart pruning method. The sentence length distribution has been integrated into Figure 3 to show that the predominant part of our real-world corpus can be processed using this information-rich method with very modest parse times.

The large amount of short inputs is at first surprising, moreover as most of these inputs can not be parsed, as can be seen in Figure 5. The explanation

is easy: most of these inputs are non-sentences such as headings, enumerations, footnotes and such. How we deal with this kind of input will be described in the section about fragmentary input.

All measurements were carried out on an Intel XEON E5430 2.66GHz cluster computer. Except for the parallelization, the used hardware equals a modern standard desktop PC, which again shows the feasibility of the used method.



Mean parse time (CPU sec) over sentence length

	No pruning	Max. 400 passive	Max. 100 passive
Avg. Parse Time (CPU sec)	5.90	3.95	2.17
Unparsed Sentences	433104 (28.2%)	392758 (25.5%)	381019 (24.8%)
Recall	71.8%	74.5%	75.2%
Best Parse Lost		5.43%	19.7%

Fig. 4. Comparison of results with different chart pruning settings

Figure 4 shows the effects of the chart pruning approach using moderate as well as more aggressive pruning. The last row displays the amount of parsed sentences which do not get the best results due to pruning. Note that the increase in parsed sentences is only due to the reduced resource needs through pruning, and that the lexical failures are not contained in the unparsed sentences figures.

Figure 5 shows the amount of unparsed sentences, split into two categories. The dots represent the sentences that could not be parsed due to time limitations, the solid lines those that were rejected by the grammar. Not surprisingly, the fraction of sentences hitting the time bound increases noticeably for sentences

longer that 60 words, but it should be noted that the percentage that can not be parsed because of grammatical reasons stays almost constant.

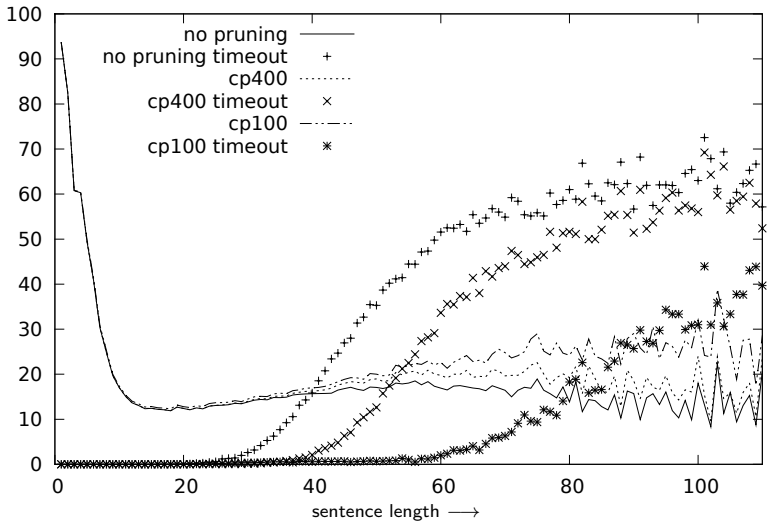


Fig. 5. Percentage of unparsed sentences over sentence length

For sentences with less than 40 words, aggressive chart pruning loses parses (around 0.8%) that the mild pruning still does successfully, because edges needed for a full parse are pruned from the chart. In toto, the aggressive pruning gets more readings because it greatly improves recall on the longer sentences, but some are lost in the important middle range, which is also why we use the results from the mild pruning for the extraction of the semantics. An advanced system could adapt pruning to the input length, or try to come up with better local models that minimize the loss of useful subconstituents.

We also compared the (absolute) scores of the discriminative model for the two variants. While the method without chart pruning always finds the best parse, this is not true for the pruned chart. The result is displayed in the fourth row of the table in Figure 4. Since the scores of the maximum entropy model are not probabilities, we can not give meaningful numbers on the loss of quality, but a rough comparison of the scores suggests that in most cases the penalty is minor.

Fragmentary Input. There are several alternatives to deal with input like headings and footnotes, one to identify and handle them in a preprocessing step, another to use a special root condition in the deep analysis component that is able to combine phrases with well-defined properties for inputs where no spanning result could be found.

We employed the second method, which has the advantage that it handles a larger range of phenomena in a homogeneous way. Figure 6 shows the change in percentage of unparsed and timed out inputs for the mild pruning method with and without the root condition combining fragments.

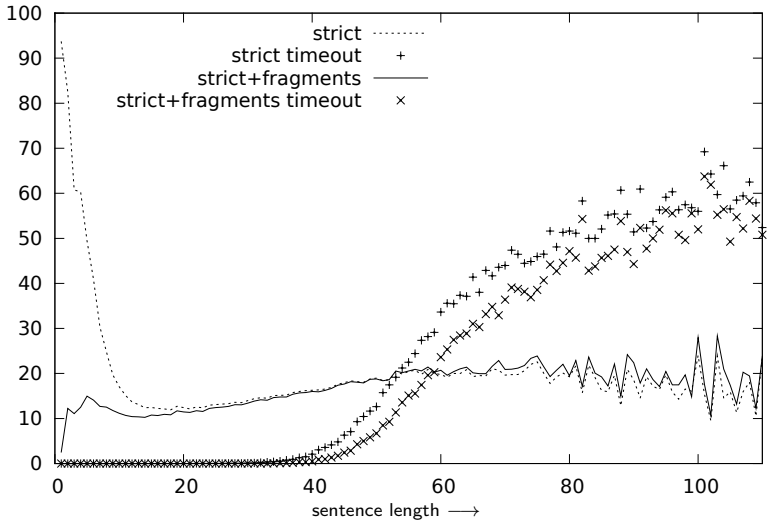


Fig. 6. Unparsed and timed out sentences with and without fragment combination

As Figure 6 shows nicely, this changes the curve for unparsed sentences towards more expected characteristics and removes the uncommonly high percentage of short sentences for which no parse can be found.

Together with the parses for fragmented input, we get a recall (sentences with at least one parse) over the whole corpus of 85.9% (1,321,336 sentences), without a significant change for any of the other numbers.

2.6 Parser Output

In contrast to shallow parsers, the ERG not only handles detailed syntactic analyses of phrases, compounds, coordination, negation and other linguistic phenomena that are important for extracting relations, but also generates a formal semantic representation of the meaning of the input sentence in the MRS representation format (Minimal Recursion Semantics; [6]). It is comparable to a first order logic form. It consists of so-called elementary predications for each token and larger constituents, connected via argument positions and variables/labels, from which the predicate-argument structure can be derived (example in Figure 7).

$$
\begin{array}{l}
\langle\, h_1, \\
\quad h_3{:}\mathsf{udef\_q}(x_5\{\mathsf{PERS}\ 3, \mathsf{NUM}\ sg\},\ h_4,\ h_6), \\
\quad h_7{:}\mathsf{\_semantic\_a\_1}(e_8\{\mathsf{SF}\ prop, \mathsf{TENSE}\ untensed, \mathsf{MOOD}\ indicative\},\ x_5), \\
\quad h_7{:}\mathsf{\_similarity\_n\_to}(x_5,\ i_9), \\
\quad h_{10}{:}\mathsf{\_measure\_v\_1}(e_2\{\mathsf{SF}\ prop, \mathsf{TENSE}\ pres, \mathsf{MOOD}\ indicative, \mathsf{PROG}\ \text{-}, \mathsf{PERF}\ \text{-}\},\ p_{11},\ x_5), \\
\quad h_{10}{:}\mathsf{parg\_d}(e_{12}\{\mathsf{SF}\ prop\},\ e_2,\ x_5), \\
\quad h_{10}{:}\mathsf{\_in\_p}(e_{13}\{\mathsf{SF}\ prop, \mathsf{TENSE}\ untensed, \mathsf{MOOD}\ indicative\},\ e_2,\ x_{14}\{\mathsf{PERS}\ 3, \mathsf{NUM}\ pl, \mathsf{IND}\ +\}), \\
\quad h_{15}{:}\mathsf{udef\_q}(x_{14},\ h_{16},\ h_{17}), \\
\quad h_{18}{:}\mathsf{\_term\_n\_of}(x_{14},\ x_{19}\{\mathsf{PERS}\ 3, \mathsf{NUM}\ pl\}), \\
\quad h_{20}{:}\mathsf{udef\_q}(x_{19},\ h_{21},\ h_{22}), \\
\quad h_{23}{:}\mathsf{compound}(e_{25}\{\mathsf{SF}\ prop, \mathsf{TENSE}\ untensed, \mathsf{MOOD}\ indicative, \mathsf{PROG}\ \text{-}, \mathsf{PERF}\ \text{-}\},\ x_{19},\ x_{24}), \\
\quad h_{26}{:}\mathsf{udef\_q}(x_{24},\ h_{27},\ h_{28}), \\
\quad h_{29}{:}\mathsf{\_similar\_a\_to}(e_{30}\{\mathsf{SF}\ prop, \mathsf{TENSE}\ untensed, \mathsf{MOOD}\ indicative\},\ x_{24}), \\
\quad h_{29}{:}\mathsf{comp}(e_{32}\{\mathsf{SF}\ prop\},\ e_{30},\ u_{31}), \\
\quad h_{29}{:}\mathsf{\_word\_n\_of}(x_{24},\ i_{33}), \\
\quad h_{23}{:}\mathsf{\_context\_n\_1}(x_{19}) \\
\quad \{\, h_{27} =_q h_{29},\ h_{21} =_q h_{23},\ h_{16} =_q h_{18},\ h_4 =_q h_7 \,\} \,\rangle
\end{array}
$$

Fig. 7. Sample MRS for the sentence "Semantic similarity is measured in terms of similar word contexts."

As in previous work [18] and because of the increased parsing recall, we again opt for precision and only use results from the deep parser instead of extending the hybrid workflow (Figure 2) in such a way that a shallow parser with less detailed analyses is used as fall-back in case deep parsing fails (as done in an intermediate system, [17]).

3 Application: Semantic Search Based on Extracted Predicate-Argument Structure

The idea of the semantic search application is to use the sentence-wise semantic representations generated offline by the deep parser. From its output, a normalized predicate-argument structure is extracted that is stored in a search index. The main motivation is at least partial abstraction from syntactic variants. Thus, the extraction process includes dividing sentences with coordination into independent structures, and using the semantic subject and object in both active and passive sentence construction independently of the syntactic realization.

The user interface for this application is simple. Instead of a single search text input field, the user will see three: one for subject, one for predicate and another one for further objects. This is easy to understand also for non-linguists, and fields may be left emtpy to match anything. In the current version, the search interface supports the use of synsets of predicates only.

3.1 Extracting Predicate-Argument Structure from MRS

The MRS representations resulting from hybrid parsing are relatively close to linguistic structures and contain more detailed information than a user would

like to query and search for. Therefore, an additional extraction and abstraction step is necessary before storing the semantic structures in the search index.

The format we devised for this purpose we call *semantic tuples*, a blend of triples and quintuples, as we store quintuples (subject, predicate, direct object, other complements and adjunct), but to ease search term input for the user, only distinguish between a triple of subject, predicate and any other objects in the query structure.

The algorithm to generate the semantic tuples first performs an intermediate transformation into isomorphic, serializable Java objects that can be made persistent. On these objects, efficient graph manipulation resulting in extracted semantic tuples can take place. Handling of coordination has been implemented by generating multiple tuples. Passive constructions are elegantly handled by the grammar itself and lead to identical semantic tuples regardless of active or passive formulation of the same proposition.

Due to semantic ambiguity, the deep parser may return more than one reading per sentence. Currently up to three readings are considered (the most probable ones according to the treebank-trained parse ranking model), and semantic tuples are generated for each reading respectively. Multiple readings may collapse into the same semantic tuple structure, in which case only a single one is stored in the database. Otherwise, a voting mechanism based on rank and number of isomorphic semantic tuples decides for the best selection.

The following sentence includes the semantic tuple structure (in brackets):

"[We]$_{\text{SUBJ}}$ [evaluate]$_{\text{PRED}}$ [the efficiency and performance]$_{\text{DOBJ}}$ [against the corpus]$_{\text{ADJU}}$."

In this example, the conjunction relation connects two noun phrases, both of them being DOBJ; therefore, no new semantic tuple is necessary. However, we decided to distinguish cases where conjunction connects two sentences or verb phrases. In such cases, semantic tuples are generated for each part respectively. The following example shows an *AND* relation. Conjunction relations may also be realized in different lexemes, e.g. *and, but, or, as well as*, etc.

For the sentence *"The system automatically extracts pairs of syntactic units from a text and assigns a semantic relation to each pair."*, two semantic tuples are generated separately with their own PRED, DOBJ and OCMP:

"[The system]$_{\text{SUBJ}}$ [extracts]$_{\text{PRED}}$ [pairs of syntactic units]$_{\text{DOBJ}}$ [from a text]$_{\text{OCMP}}$ [automatically]$_{\text{ADJU}}$."

and

"[The system]$_{\text{SUBJ}}$ [assigns]$_{\text{PRED}}$ [a semantic relation]$_{\text{DOBJ}}$ [to each pair]$_{\text{OCMP}}$ [automatically]$_{\text{ADJU}}$."

In passive sentences, the syntactic subject becomes the semantic object and vice versa:

"[Unseen input]$_{\text{DOBJ}}$ [was classified]$_{\text{PRED}}$ [by trained neural networks with varying error rates depending corpus type]$_{\text{SUBJ}}$."

3.2 Filling the Search Index

For each sentence, the semantic tuple structure together with associated character span information relative to the sentence start is then stored in an Apache Solr[5] search index. It also contains metainformation on page number, sentence number, offset and document ID.

In case a named entity is identified by the named entity recognizer, further information on span and type (such as location, person, time) of the item is stored. This named entity type information is used to identify the answer candidate type in an additional question answering interface we will not further describe in this paper. The following snippet from Solr input for a single sentence may give an impression of the underlying index schema.

```
<doc>
  <field name="aclaid">N07-1043</field>
  <field name="page">2</field>
  <field name="sentno">56</field>
  <field name="prefix">N07-1043-s56-p2</field>
  <field name="offset">353</field>
  <field name="qgen">PET</field>
  <field name="sentence">Sahami et al., (2006) measure semantic
     similarity between two queries using the snippets returned
     for those queries by a search engine.</field>
  <field name="subj">Sahami 2006 et al.</field>
  <field name="subj_start">0</field>
  <field name="subj_end">12</field>
  <field name="pred">measure</field>
  <field name="pred_start">22</field>
  <field name="pred_end">28</field>
  <field name="dobj">semantic similarity</field>
  <field name="dobj_start">30</field>
  <field name="dobj_end">48</field>
  <field name="ocmp">between two queries using the snippets
     returned for those queries by a search engine</field>
  <field name="ocmp_start">0</field>
  <field name="ocmp_end">133</field>
  <field name="ner_types">citation ne-term ne-term </field>
  <field name="ner_cstart">0 30 121 </field>
  <field name="ner_cend">20 48 133 </field>
  <field name="ner_surface">"Sahami et al., (2006)"
                            "semantic similarity"
                            "search engine" </field>
</doc>
```

To sum up the overall offline analysis for search index generation, Figure 8 depicts the offline NLP and semantic tuple extraction workflow.

[5] http://lucene.apache.org/solr

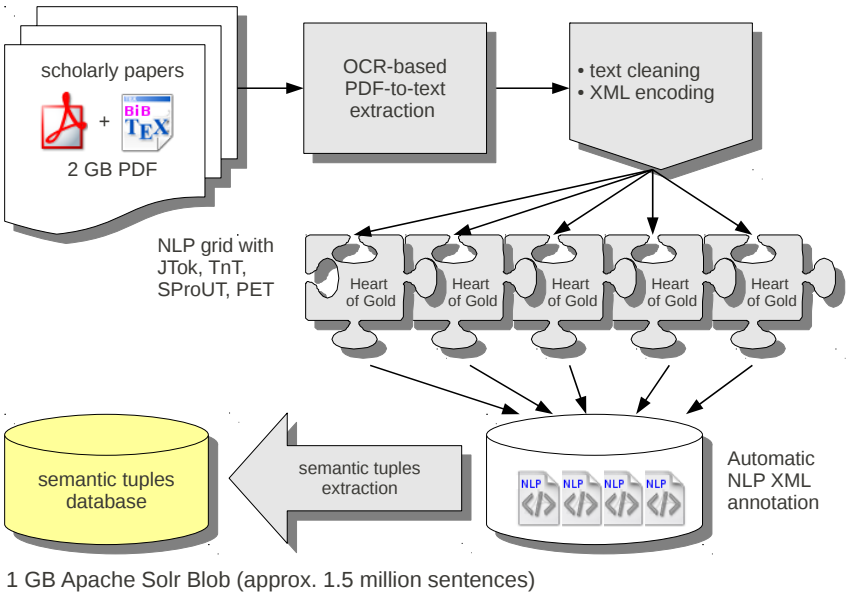


Fig. 8. Grid-based hybrid parsing of the scientific paper corpus

3.3 Query Interface

As depicted in Figure 9, the user interface for semantic paper search contains three text fields where the user can input subject, predicate and all remaining structures (rest). The latter is combined to ease input (otherwise users would become worried about what to put in OCMP or ADJU) and will be expanded to a disjunctive Solr/Lucene query expression.

Fig. 9. Simple query interface

To give an example, a semantic tuple search expression with input to field subject=*, input to field predicate='measure', and input to field rest='semantic similarity' is translated into an Apache Solr query

```
pred:measure +(dobj:"semantic similarity"
```

```
                    OR ocmp:"semantic similarity"
                    OR adju:"semantic similarity")
```

In case WordNet synset [10] expansion is enabled, `measure` is replaced by `(measure OR evaluate OR quantify OR value OR assess OR valuate)`.

It is planned to also allow for synonym search in the SUBJ and REST field. Here, domain ontology information as well as automatically identified similar (multi-word) terms could be used to expand the query.

Search Results for * "measure" "semantic similarity"

- N07-1043: Sahami et al., (2006) [measure]$_{\text{PRED}}$ [semantic similarity]$_{\text{DOBJ}}$ between two queries using the snippets returned for those queries by a search engine.
- W04-0106: [Semantic similarity]$_{\text{DOBJ}}$ [is measured]$_{\text{PRED}}$ in terms of similar word contexts.
- N07-1044: [The semantic similarity]$_{\text{DOBJ}}$ between neighbors and senses [is measured]$_{\text{PRED}}$ using a manually crafted taxonomy such as WordNet (see Budanitsky and Hirst 2001 for an overview of WordNet-based similarity measures).
- P08-1028: We [assessed]$_{\text{PRED}}$ [a wide range of semantic similarity measures]$_{\text{DOBJ}}$ using the WordNet similarity package (Pedersen et al., 2004).
- W06-3802: Using WordNet, we [can measure]$_{\text{PRED}}$ [the semantic similarity]$_{\text{DOBJ}}$ or relatedness between a pair of concepts (or word senses), and by extension, between a pair of sentences.
- W06-1659: Using WordNet, we [can measure]$_{\text{PRED}}$ [the semantic similarity]$_{\text{DOBJ}}$ or relatedness between a pair of concepts (or word senses), and by extension, between a pair of sentences.
- W05-1203: For entailment identification, since this is a directional relation, we [only measure]$_{\text{PRED}}$ [the semantic similarity]$_{\text{DOBJ}}$ with respect to the hypothesis (the text that is entailed).
- W06-1104: We [measured]$_{\text{PRED}}$ [semantic relat-edness instead of semantic similarity]$_{\text{DOBJ}}$.
- P06-1112: 3. [The semantic similarity SemSim(h , h)]$_{\text{DOBJ}}$ [is measured]$_{\text{PRED}}$ using Word-Net and eXtended WordNet.

...

Fig. 10. The first matching sentences in the ACL Anthology subset 2002-2008 with recognized variation in predicate synsets (assess, measure, evaluate) and passive constructions

The result is then a list of sentence snippets (Figure 10). By clicking on a hyperlink underlying the snippet text, the original PDF is opened. By using the information on page and sentence text/offset in the Apache Solr answer, the result sentence is highlighted as shown in Figure 11. This helps to quickly identify relevance of the answer by looking at context in the original layout.

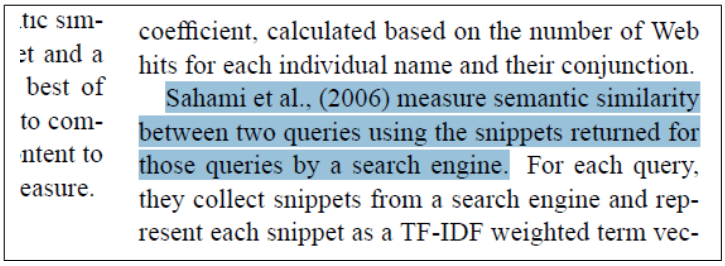


Fig. 11. First result sentence (from N07-1043) highlighted in original PDF

4 Related Work

Using HPSG combined with shallow domain-specific modeling for high-precision analysis of scientific texts is an emerging research area. Another ERG-based approach to relation and information extraction from scientific texts is SciBorg [13]. SciBorg mainly deals with chemistry research papers and handles domain-specific phenomena with a specialized named entity recognizer. It relies on a shallow parser as robustness fall-back for MRS generation.

Other groups use less elaborated and fine-grained HPSG grammars than ERG. [11] report on large-scale parsing of MEDLINE articles (1.4 billion words) with such a simplified grammar.

[14] use shallow dependency structure and results from HPSG parsing for extracting protein-protein interactions (PPI) from research papers. The same group has also worked on medical texts: MEDIE[6] is a semantic search engine to retrieve biomedical correlations from MEDLINE articles.

What distinguishes our approach from those, besides concentration on a different scientific area, is the focus on and use of ontology information as integrated part of linguistic analysis, use of the most comprehensive and elaborated HPSG grammar for English (ERG), and the interactive user interface (Scientist's Workbench application; [17]) and editor [18].

5 Conclusion and Future Work

We have presented our recent advances in full, robust parsing of scientific papers texts. By careful preprocessing and novel approaches to efficient parsing of long sentences, we could improve coverage from 65 to more than 85%.

The semantic search application built on the semantic representations generated by the deep grammar is a useful extension to cope with synonyms and syntactic variation when querying full scientific publication content. The search space, initially expanded by adding synonymns, can be again constrained by imposing semantic subject-predicate-object structure in the query.

[6] `http://www-tsujii.is.s.u-tokyo.ac.jp/medie/`

Further research goals are improving robustness of the NLP tool chain. We are also working on generic techniques to automatically extract and use science domain information from the underlying paper corpus to improve targeted search. Three main tasks in our focus are coreference resolution, term extraction and ontology extraction viz. population. The idea is that these techniques, in a first step gained independently from the text corpus or partially from NLP analyses of it, will benefit from each other and can be used to build more reliable and precise resources and tools in a bootstrapping process.

Handling of negation, modal constructions, subclauses etc. also fall into the category deep NLP can handle, but this will be addressed in the future as it also requires lexico-semantic information of verbs etc. in the extraction process. It will definitely be an important extension helping to improve precision in search.

The semantic search application is part of the Scientist's workbench and is complemented by a visualization and navigation tool TeeCeeGeeNav [16] that supports scientists in quickly getting an overview of a (new) research field by browsing through a typed citation graph computed from the scientific paper corpus. The citation classification with categories such as use or refutation of results of the cited paper currently builds on shallow NLP (such as PoS tagging) only. In the future, deep semantics could help too further improve this difficult classification task.

Acknowledgments

The authors would like to thank Peter Adolphs, Dan Flickinger and Stephan Oepen for their support and numerous fruitful discussions. We would also like to thank Yi Zhang and Bart Cramer for the implementation of chart pruning in PET and their support to put it into use. The work described in this paper has been carried out in the context of the project TAKE (Technologies for Advanced Knowledge Extraction), funded under contract 01IW08003 by the German Federal Ministry of Education and Research, and in the context of the world-wide DELPH-IN collaboration[7].

References

1. Adolphs, P., Oepen, S., Callmeier, U., Crysmann, B., Flickinger, D., Kiefer, B.: Some fine points of hybrid natural language parsing. In: Proc. of LREC. pp. 1380–1387. Marrakesh, Morocco (2008)
2. Bird, S., Dale, R., Dorr, B., Gibson, B., Joseph, M., Kan, M.Y., Lee, D., Powley, B., Radev, D., Tan, Y.F.: The ACL anthology reference corpus: A reference dataset for bibliographic research. In: Proc. of LREC. pp. 1755–1759. Marrakesh, Morocco (2008)
3. Brants, T.: TnT – A Statistical Part-of-Speech Tagger. In: Proc. of ANLP-2000. pp. 224–231. Seattle, WA (2000)

[7] DEep Linguistic Processing with Hpsg INitiative; `http://www.delph-in.net`

4. Callmeier, U.: PET – A platform for experimentation with efficient HPSG processing techniques. Natural Language Engineering 6(1), 99–108 (2000)
5. Copestake, A., Flickinger, D.: An open-source grammar development environment and broad-coverage English grammar using HPSG. In: Proc. of LREC. pp. 591–598. Athens, Greece (2000)
6. Copestake, A., Flickinger, D., Sag, I.A., Pollard, C.: Minimal recursion semantics: an introduction. Research on Language and Computation 3(2–3), 281–332 (2005)
7. Cramer, B., Zhang, Y.: Constraining robust constructions for broad-coverage parsing with precision grammars. In: Proc. of COLING. pp. 223–231. Beijing, China (2010)
8. Drożdżyński, W., Krieger, H.U., Piskorski, J., Schäfer, U., Xu, F.: Shallow processing with unification and typed feature structures – foundations and applications. Künstliche Intelligenz 2004(1), 17–23 (2004)
9. Flickinger, D., Oepen, S., Ytrestøl, G.: WikiWoods: Syntacto-semantic annotation for English Wikipedia. In: Proc. of LREC. pp. 1665–1671. Valletta, Malta (2010)
10. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Five papers on WordNet. Tech. rep., Cognitive Science Laboratory, Princeton University (1993)
11. Ninomiya, T., Tsuruoka, Y., Miyao, Y., Taura, K., Tsujii, J.: Fast and scalable HPSG parsing. Traitement automatique des langues (TAL) 46(2) (2006)
12. Pollard, C., Sag, I.A.: Head-Driven Phrase Structure Grammar. Studies in Contemporary Linguistics, University of Chicago Press, Chicago (1994)
13. Rupp, C., Copestake, A., Corbett, P., Waldron, B.: Integrating general-purpose and domain-specific components in the analysis of scientific text. In: Proc. of the UK e-Science Programme All Hands Meeting 2007. Nottingham, UK (2007)
14. Sætre, R., Kenji, S., Tsujii, J.: Syntactic features for protein-protein interaction extraction. In: Baker, C.J., Jian, S. (eds.) Short Paper Proc. of the 2nd Int. Symp. on Languages in Biology and Medicine (LBM 2007). pp. 6.1–6.14. Singapore (2008)
15. Schäfer, U.: Middleware for creating and combining multi-dimensional NLP markup. In: Proc. of the EACL-2006 Workshop on Multi-dimensional Markup in Natural Language Processing. pp. 81–84. Trento, Italy (2006)
16. Schäfer, U., Kasterka, U.: Scientific authoring support: A tool to navigate in typed citation graphs. In: Proc. of the NAACL-HLT 2010 Workshop on Computational Linguistics and Writing. pp. 7–14. Los Angeles, CA (2010)
17. Schäfer, U., Spurk, C.: TAKE Scientist's Workbench: Semantic search and citation-based visual navigation in scholar papers. In: Proc. of the 4th IEEE Int. Conference on Semantic Computing (ICSC-2010). pp. 317–324. Pittsburgh, PA (2010)
18. Schäfer, U., Uszkoreit, H., Federmann, C., Marek, T., Zhang, Y.: Extracting and querying relations in scientific papers. In: Proc. of the 31st Annual German Conference on Artificial Intelligence (KI-2008). pp. 127–134. Springer LNAI 5243 (2008)