Constructing Large Proposition Databases

Peter Exner **Pierre Nugues**

Lund University
Department of Computer science
Lund, Sweden
Peter.Exner@cs.lth.se, Pierre.Nugues@cs.lth.se

Abstract

With the advent of massive online encyclopedic corpora such as Wikipedia, it has become possible to apply a systematic analysis to a wide range of documents covering a significant part of human knowledge. Using semantic parsers, it has become possible to extract such knowledge in the form of propositions (predicate–argument structures) and build large proposition databases from these documents. This paper describes the creation of multilingual proposition databases using generic semantic dependency parsing. Using Wikipedia, we extracted, processed, clustered, and evaluated a large number of propositions. We built an architecture to provide a complete pipeline dealing with the input of text, extraction of knowledge, storage, and presentation of the resulting propositions.

Keywords: Semantic Parsing, Ranking Algorithm, Proposition Database

1. Introduction

With the advent of massive online encyclopedic corpora such as Wikipedia, it has become possible to apply a systematic analysis to a wide range of documents covering a significant part of human knowledge. Using semantic parsers or related techniques, it has become possible to extract such knowledge in the form of propositions (predicate–argument structures) and build large proposition databases from these documents.

While most approaches focus on shallow analysis and do not capture the full meaning of a sentence, semantic parsing goes deeper and discovers more information from text with a higher accuracy. Christensen et al. (2010) showed that using a semantic parser in information extraction can yield a higher precision and recall in areas where shallow syntactic approaches had failed. This deeper analysis can be applied to discover temporal and location-based propositions from documents.

The accuracy of semantic parsers comes at a higher cost in terms of execution time. However, in the recent years, statistical parsing and especially semantic parsing have become increasingly time-efficient in analyzing text, while maintaining superior accuracy.

This paper describes the creation of multilingual proposition databases using generic semantic dependency parsing. Using a broad domain encyclopedic corpus, Wikipedia, we extracted, processed, clustered, and evaluated a large number of propositions. We built an architecture to provide a complete pipeline dealing with the input of text, extraction of knowledge, storage, and presentation of the resulting propositions. Furthermore, our system is able to handle large-scale extractions, wide domains, and multiple input languages. Wherever possible, the handling of information is automated such that manual labor is kept to a minimum.

We believe proposition databases like the one we constructed, combined with other lexical databases, can be key components in semantic search technology, machine translation, and question answering (QA) systems.

2. Extracting Propositions

The creation of a proposition bank can be achieved through the manual annotation of a corpus (Palmer et al., 2005) or the application of an automatic parser (Banko et al., 2007). In this paper, we focus on the creation of a proposition database using generic semantic dependency parsing. We built a system that provides a complete pipeline from the input of text, the extraction of knowledge, to the storage and presentation of the extracted propositions. The manual handling of information is minimized by automating the data flow between the subcomponents of the system. Furthermore, the system is able to handle large-scale extractions, wide domains, and multiple input languages.

Wikipedia is a popular reference work covering a large array of topics and articles written in multiple languages. To create a proposition bank with high-quality propositions, we designed a ranking algorithm that assigns scores based on the redundancy of the propositions. We carried out this work in four main steps, whose goal was to:

- Construct a semantic parsing framework to scale to large heterogeneous corpora (i.e. corpora ranging from 100,000 to a few million articles).
- Parse a substantial part of Wikipedia and create large, semantically annotated, and multilingual proposition databases.
- Create a ranking algorithm that extracts high-quality propositions.
- Construct an interface to query the proposition database.

3. System Architecture

We created a framework for multilingual proposition extraction including both English and Chinese corpora. The framework uses a complete semantic parsing pipeline and modular language models, where new languages can be added without the need of reworking extraction algorithms

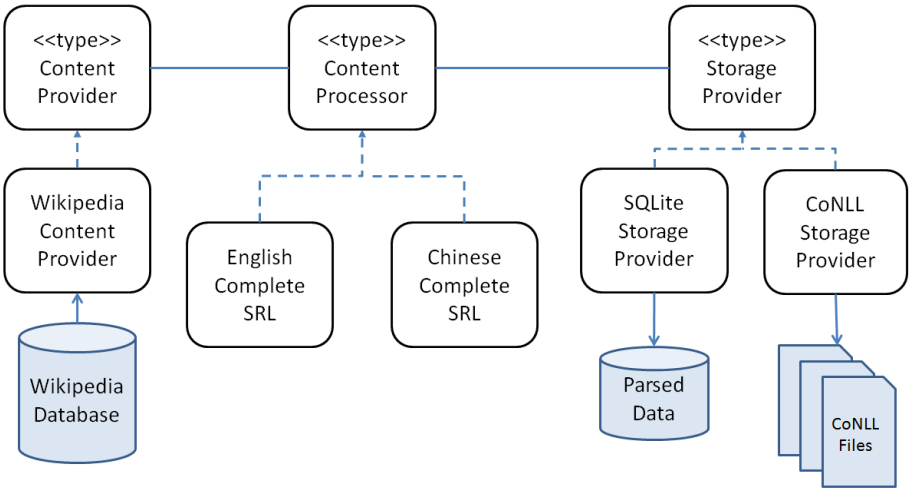

Figure 1: An overview of the parser.

or patterns. Figure 1 shows an overview of the parsing framework.

The parsing pipeline is supplied with content that can easily be extended to various corpora. The Wikipedia content provider reads articles from a Wikipedia database and then uses a language specific filter to remove markups and other items that would otherwise impede the process of parsing. Wikipedia is available in the form of XML dump files[1] provided by the Wikimedia Foundation. Although XML is a suitable format for sharing data, it is less suitable for searching a certain element within the file. For this purpose, we developed a converter that takes a Wikimedia XML file and converts it to a SQLite[2] database. We have then a fast random access to any article, something that would otherwise not be possible using only the XML dump file. The database also allows for the storage and individual updating of the articles. The parsed output from the pipeline is stored by a storage provider. We created two storage providers, a SQLite database and a CoNLL file storage provider, for the end storage of the semantically parsed text.

We wrote a server that communicates with computing nodes and launches parsing jobs on the given nodes. The server and the computing nodes communicate through a message system similar to the message passing interface (MPI) API. The server accepts a desired range of Wikipedia article identifiers. These are then subdivided by the server into suitable subranges and distributed among the computing nodes. Each node uses a complete pipeline, performing all functions from filtering to semantic annotation, to parse an article. After completion, the parsed article is sent back to the server and a new article is assigned to the computing node. This is repeated until all the desired documents have been parsed.

Using this parsing framework, we have parsed more than 30% of the English Wikipedia in approximately 4 weeks on a cluster of 10 machines. The statistics generated from this data are vital in determining the focus for our efforts and also the approach for creating the ranking algorithm. Table 1 shows an overview of the number parsed articles, sentences, and propositions.

English Wikipedia	
Articles	1,157,054
Sentences	23,754,110
Propositions	93,040,920

Table 1: An overview of parsing statistics.

4. Semantic Parser

The content processor uses a high-performance multilingual semantic parser (Björkelund et al., 2010). This parser reached high scores in the CoNLL 2009 (Hajič et al., 2009) shared task, has fast processing time, and the code is open source and freely available. The English data models used in our parser have been created from the corpus provided in the CoNLL 2008 (Surdeanu et al., 2008) shared task. The CoNLL 2008 corpus used for training is based on an annotated version of the Wall Street Journal, it is thus limited to a narrow domain. The Chinese data models have been created from a semantically annotated Chinese Treebank (Palmer and Xue, 2009).

5. Proposition Database

The proposition database is used for storing parsed data, retrieving statistics, and building Lucene indexes for the querying interface. It provides a unified schema for storing and retrieving propositions. This schema is designed to handle semantically annotated sentences as defined by CoNLL 2008 (Surdeanu et al., 2008). It is in this sense a generic structure capable of handling parsed data from more than one parsing configuration. It is also suitable for providing simple statistics, such as the number of propositions, by means of SQL queries. Figure 2 shows the data model.

The proposition database also features a simple API, which allows the creation of databases, as well as storing and re-

[1] http://en.wikipedia.org/wiki/Wikipedia:Database_download
[2] http://www.sqlite.org/

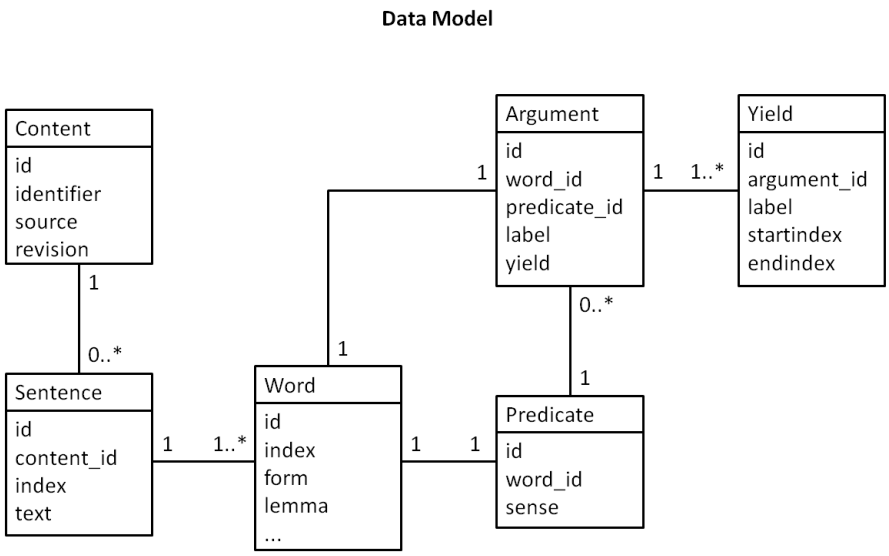

Figure 2: An overview of the data model

trieving propositions. The API makes good use of transactions, ensuring data integrity by making sure that parsed content is stored in its entirety.

The database aggregator assembles the many smaller databases created by the parsing jobs into one large database. The smaller databases are read one by one from a folder and added to the final database. This large database is more suitable for data processing tasks such as retrieving statistics and building the Lucene index.

For distribution purposes, we also store the parsed articles in individual files using CoNLL 2008 annotation. Since the CoNLL 2008 annotation has a large recognition and use in NLP tools, we believe that by providing the parsed Wikipedia articles in this format we encourage and facilitate the reuse of data.

6. Ranking Algorithm

The occurrence of erroneous extractions is a problem found in all extraction systems. In order to filter out less likely propositions, we have developed a ranking algorithm based on the redundant occurrences of propositions in text.

Our ranking algorithm assigns a score to propositions based on their redundancy. Propositions are considered to be redundant if more than one proposition has the same predicate and its arguments have the same headword. We create the score by dividing the number of redundant propositions by the total number of propositions for a certain predicate. This score is then assigned to the propositions having that predicate.

As an example, consider the data shown in Table 2. We have two tuples with redundancy #1 and #2, together they have (5+2) = 7 propositions. In all, there are (5+2+1) = 8 propositions for the predicate, *describe*. This gives a redundancy score of 7 / 8 = 87.5%.

This algorithm can be used for ranking semantic searches and also to create new corpora containing higher quality propositions. An overview of the types and number of redundant propositions created by our ranking algorithm based on 10% of parsed data can be seen in Table 3

Although our algorithm assigns a score to only a small subset of propositions, we believe a higher yield can be achieved through the use of a coreference solver and other lexical databases.

#	Argument 1	Predicate	Argument 2	Count
1	equations	describe.01	laws	5
2	methods	describe.01	approach	2
3	papers	describe.01	algorithm	1

Table 2: An example of predicate argument distribution where #1 and #2 are redundant and #3 is a hapax.

Type	Distribution
All propositions with two arguments	54.9%
Propositions with redundancy	29.2%
All (Noun, Verb, Noun) Propositions	7.8%
(Noun, Verb, Noun) Propositions without redundancy	6.0%
(Noun, Verb, Noun) Propositions with redundancy	1.7%
(Noun, Verb, Noun) Unique propositions with redundancy	0.4%

Table 3: The number of propositions grouped by proposition type, based on 10% of parsed data.

7. Querying Interface

We developed a web-based query interface to the proposition databases. The interface allows the use of temporal and location based searches. This makes use of the semantic properties of the proposition database and creates new possibilities in semantic search.

Figure 3 shows an example of a search. It is possible to search for propositions from the English and Chinese Wikipedia. Searches are made by entering the predicate and arguments in lexical form. For instance, to search for *who built the pyramids*, one enters the lemmatized form of *built*, *build*, into the predicate field. Figure 4 shows the results of the query, the arguments in the sentences are colored differently depending on their semantic roles.

The query interface to the proposition databases is available from this location: `http://barbar.cs.lth.se:8071/`

8. Conclusion & Application

In this paper, we described an end–to–end framework for extracting, storing, ranking, and querying predicate–argument structures from large heterogeneous corpora. We implemented a parsing framework, capable of performing parallel extraction on multiple computing nodes. Using this framework we parsed 30% of the English Wikipedia, extracted about 93,000,000 predicate–argument structures and stored them in a proposition database. We also explored a ranking algorithm that scores propositions based on their redundancy. Applied to a subset of extracted propositions, we believe the ranking algorithm can be used in ranking semantic searches and also to create new corpora containing higher quality propositions.

We believe that the ranking algorithm could be improved using a coreference solver that would tie pronouns such as she, he, or it to person or organization names. In the future, we also plan to parse the complete Wikipedia corpus in English and other languages.

The resulting proposition database has been used in a separate project to investigate the use of semantic parsing to extract events from text (Exner and Nugues, 2011). By using predicate-–argument structures extracted from 10% of the English Wikipedia and a converter using VerbNet thematic roles, we produced 27,500 events in the LODE RDF format (Shaw et al., 2009).

9. Acknowledgments

This research was supported by Vetenskapsrådet, the Swedish research council, under grant 621-2010-4800, the European Union's seventh framework program (FP7/2007-2013) under grant agreement N^{O} 230902, and the eSSENCE research program.

10. References

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In Manuela M. Veloso, editor, *IJCAI*, pages 2670–2676.

Anders Björkelund, Bernd Bohnet, Love Hafdell, and Pierre Nugues. 2010. A high-performance syntactic and semantic dependency parser. In *COLING (Demos)*, pages 33–36. Demonstrations Volume.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2010. Semantic role labeling for open information extraction. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, FAM-LbR '10, pages 52–60.

Peter Exner and Pierre Nugues. 2011. Using semantic role labeling to extract events from wikipedia. In *Proceedings of the Workhop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011)*, pages 38–47, Bonn.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The conll-2009 shared task: syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 1–18.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: an annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105.

Ryan Shaw, Raphaël Troncy, and Lynda Hardman. 2009. LODE: Linking open descriptions of events. In *4th Asian Semantic Web Conference*, pages 153–167, December 6-9.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The conll-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, CoNLL '08, pages 159–177.

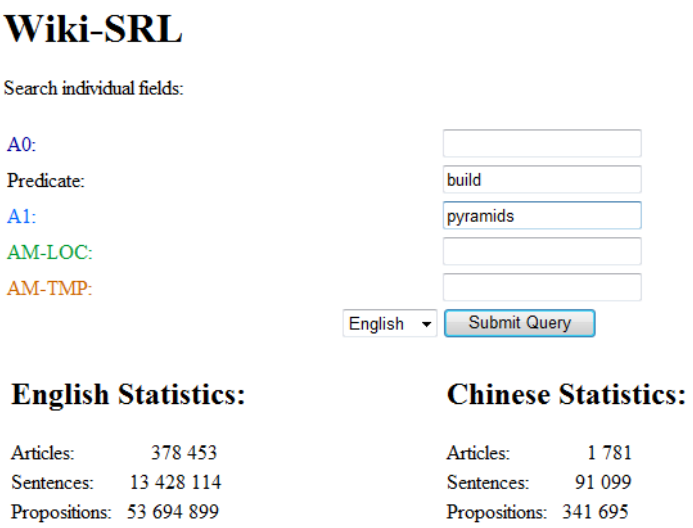

Figure 3: Searching the proposition database.

build pyramids (2)
1. **build** the pyramids$_{A1}$
2. **build** pyramids$_{A1}$
building Pyramids (3)
3. **building** the Pyramids$_{A1}$
4. **building** pyramids$_{A1}$, columns , and such structures
5. **building** pyramids$_{A1}$
built pyramids (3)
6. **built** huge pyramids$_{A1}$ and temples
7. the Giza pyramids$_{A1}$ **built**
8. Egypt 's great pyramids$_{A1}$ **built**
build pyramids atop (1)
9. **build** new temple pyramids$_{A1}$ atop$_{AM-LOC}$ older ones
acrobats **build** pyramids (1)
10. the acrobats$_{A0}$ themselves **build** human pyramids$_{A1}$
cards **build** is (1)
11. hold four Hero and/or Wonder cards$_{A0}$ **build** the Pyramids Mare Nostrum is$_{A1}$ intended by the designer to be a more playable version of Civilization
Children **build** pyramids (1)
12. Children$_{A0}$ **build** " Lambertus pyramids$_{A1}$ " of branches , decorated with lanterns and lamps around which they dance and sing traditional songs (known as Lambertussingen or Käskenspiel)
He **build** first (1)
13. He$_{A0}$ **build** the first$_{A1}$ of the pyramids , a step pyramid for him at Saqqara
humans **build** pyramids (1)
14. ancient humans$_{A0}$ **build** pyramids$_{A1}$
it **build** pyramids in (1)
15. it$_{A0}$ **build** totally useless pyramids$_{A1}$ in$_{AM-PNC}$ order to stimulate the economy , raise aggregate demand , and encourage full employment
Olmecs **build** pyramids (1)
16. The Olmecs$_{A0}$ **build** pyramids$_{A1}$
pharaohs **build** pyramids (1)
17. Middle Kingdom pharaohs$_{A0}$ **build** pyramids$_{A1}$

Figure 4: Results from searching the proposition database.