Automatic transformation of phrase treebanks to dependency trees

Michael Daum, Kilian A. Foth, Wolfgang Menzel
Natural Language Systems
Department of Computer Science
University of Hamburg

(micha|foth|wolfgang)@nats.informatik.uni-hamburg.de

Abstract

Word-to-word dependency structures are useful for consistent representation and comparable evaluation of parsing results. However, most large-scale treebanks contain various variants of phrase structure trees, since automatic parsers usually produce constituent structures. We present a freely available extensible tool for converting phrase structure to dependencies automatically, and discuss its application to the NEGRA treebank of German.

1. Introduction

With the rise of statistical methods in NLP, treebanks have become an important resource for various tasks: they are used to exemplify models of natural languages, to calculate probabilities of alternative generation rules, or to induce entire grammars. The most widespread use of large treebanks is in developing and testing of automatic syntactic parsers, because both the training and the evaluation of a probabilistic parsing system require large amounts of annotated syntactic structures; but they are also useful as a source of examples for the creation of entirely rule-based NLP systems and other kinds of linguistic inquiry (Bouma and Kloosterman, 2002).

Various formalisms are used in automatic syntax analysis (PCFG, LFG, TAG), but most large treebanks have so far been compiled in some form of constituent structure. However, this kind of model may actually not be ideal for comparing and evaluating the performance of automatic parsers, and it has been suggested (Lin, 1995) that dependency trees would allow for more meaningful error measures and comparisons. More recently (Beil et al., 2002) confirm the need for this kind of evaluation.

Dependency and constituent structure are different models of natural language, but they are similar enough that it is conceivable to convert one model into the other automatically. Rather than generate new treebanks that use dependencies, we argue that it is useful to convert existing data to dependency relations automatically. We describe a freely available tool that can import the Penn treebank (Marcus et al., 1994) and the NEGRA treebank (Skut et al., 1997) formats and generate dependency descriptions from them. It has been successfully used to convert the NEGRA corpus of German into dependency trees that conform to an existing dependency model of German.

2. Phrase structure and dependency structure

If we describe an utterance as an ordered sequence of words, a phrase structure tree is a set of phrases or *constituents*, each of which has a set of associated other tree elements (words or constituents). This very general definition is often extended or restricted in various ways:

- Usually, each constituent is labelled with a *category label* such as `VP` or `NP`.
- The words of an utterance are often labelled with a different set of labels corresponding to part-of-speech tags.
- The relation between a tree element and its parent constituent is sometimes labelled as well, with labels such as “Subject” or “Direct Object”.
- Often, the condition is imposed that the yields of two constituents must be either disjunct or subsets of each other, i.e. any two overlapping constituents must be properly nested. This ensures that a context-free derivation can be written to motivate each phrase tree.
- Additional elements are sometimes permitted, such as the secondary edges in the WSJ corpus, to model relations like ‘logical subject’ or anaphoric reference.

A dependency tree for the same utterance is characterized by a set of directed edges so that each word either modifies some other word or is regarded as the root of the entire tree. Normally, each edge carries a label that characterizes the relation, e.g. ‘Subject’ or ‘Object’. Usually, only properly nested structures are allowed; this is achieved by postulating the projectivity property for any pair of edges (Schröder et al., 2000), which can however be abandoned if necessary. Secondary edges can be defined much like in phrase trees to model additional relationships.

The performance of automatic parsers is usually measured by their *precision* and *recall*, defined as the ratio of computed to correctly computed and of correctly computed to annotated partial structures. Either measure can easily be increased to maximum in isolation by computing no or all possible structures; therefore both measures must always be considered in combination. Evaluation measures only differ in what they consider a correct partial result; many different approaches exist to extract and compare such substructures (Briscoe et al., 1998).

(Lin, 1995) proposed to compare annotations and parser output in dependency space and gave a general algorithm for producing dependency trees from phrase trees if necessary. In dependency space, precision and recall always coincide, since each word can only be attached correctly or not. Whereas in phrase structure space a wrong attachment

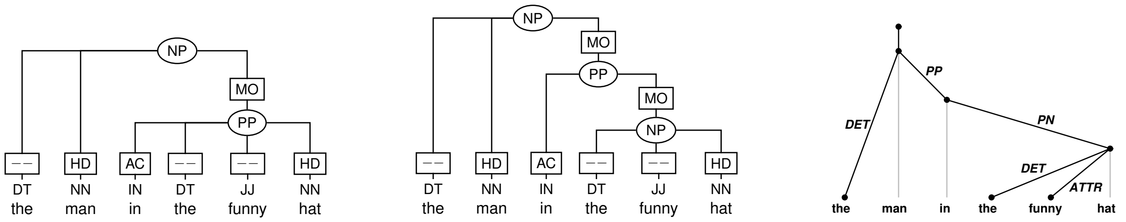

Figure 1: Ambiguous representations in phrase structure space.

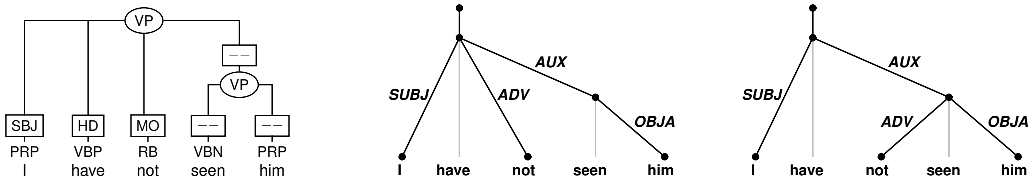

Figure 2: Ambiguous representations in dependency space.

can cause several constituents to be considered erroneous, it will always cause exactly one wrong dependency. This error measure can therefore be considered more intuitively adequate.

Dependency representation also makes it easy to evaluate the performance of a parser selectively. If edge labels are used, common tasks such as judging the performance of a parser for subject detection are solved by counting only the corresponding edges – a trivial modification of the normal evaluation method.

One other advantage of representing language as dependencies is that some gratuitous ambiguities are avoided entirely. Many of these are constructions where it is unclear whether an embedded level of phrase structure should be assumed or not (see Figure 1). For example, there is no consensus about whether particular languages actually do or do not have verb phrases; in a dependency representation this issue simply does not arise.

To be sure, there are also instances of ambiguities that arise *only* in dependency representation. Typically these are constructions where more than two constituents would form a larger phrase (see Figure 2). Lin proposes the automatic normalization of dependency trees to eliminate such inconsequential differences in an extra step.

There are also some constructions where several dependency structures are possible but none is preferable. For instance, in conjunction phrases there is no consensus about which word is the phrase head, and in fact every solution causes problems from a linguistic perspective (see Figure 3). In such cases one representation is usually chosen arbitrarily.

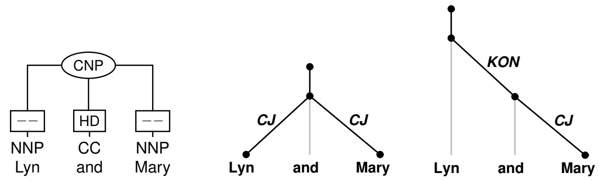

Figure 3: Different dependency structures for conjunction phrases.

3. The Tool DEPSY

Lin's proposed algorithm, taken from (Magerman, 1994), proceeds essentially as follows:

1. determine the *head child* of the root phrase node
2. apply the algorithm to all constituents of the head child, and obtain its *head word*, which will constitute the root word of the dependency tree
3. apply the algorithm to all siblings of the head child, and subordinate their head words under the current head word.

The head child of a phrase node is found by consulting a *head table* that associates each phrase category with a direction and a list of other categories; e.g. the head of a PP might be defined as the first preposition to the left, and the head of a VP as the first finite verb to the left. The resulting dependencies are simple pairs of words, with no edge labels. Also, only connected trees (where only one root node exists) can be handled.

The tool DEPSY *(Dependency Synthesizer)* implements this fundamental algorithm together with some extensions to overcome its limitations. Extending the principle of table-driven operation, the exact behaviour is controlled by several tables beyond the head table:

- a table of additional conversion functions is consulted for each node in the phrase tree before the fundamental algorithm is applied.
- a label table is used during the transformation to choose a label for each dependency edge.
- a second table of conversion functions is consulted for each word in the dependency tree after the fundamental algorithm has been applied.

As an example of a useful conversion of phrase trees, assume that PP phrases are annotated without an embedded NP in a corpus (cf. Figure 1), which means that they can have only one lexical head. The basic algorithm alone cannot derive the desired dependency structure from the left phrase tree. (Eisner, 1996) proposed inserting an embedded NP into each PP to solve this problem. By adding this conversion to our table of preprocessing functions we can determine the head of the new NP normally, without changing the basic algorithm.

As suggested by Lin, conversions on dependency trees can be defined to normalize gratuitous ambiguity. Assume that the adverbial modifiers (`MO`) of a VP are all included directly into a flat VP, but the annotation guideline for dependency trees states they should modify the full verb if possible. A conversion can be defined that moves adverbials down the auxiliary edges (`AUX`) where possible.

All of these tables and additional procedures are stored in a separate plugin that is independent of the main algorithm, so that the tool can easily be adapted different languages or different annotation styles of the same language.

DEPSY can process input in the 'merged' (`.mrg`) format of the Wall Street Journal Corpus as well as the 'export' format of the NEGRA corpus; other variants of constituent notation are simple to add. It generates a simple line-based dependency representation; an output format usable for our own dependency parser (Schröder, 2002) is also available. The entire program has been implemented in Perl for maximal portability.

4. Translating the NEGRA treebank

It is straightforward to transform a treebank into dependency structures if the only goal is to create dependency trees of some kind. However, with additional transformations it is also possible to make the output conform to an already existing modelling of the target language. A set of rules has been written that transforms the NEGRA corpus to conform closely to a pre-existing broad-coverage dependency model of German developed in the Hamburg partial parsing project (Daum et al., 2003).

Many of the transformations are obvious, such as turning subject constituents (edge label `SB`) into dependencies labelled `SUBJ`. Others require the application of additional conversions before or after the basic algorithm. Conversions equivalent to all three examples of the previous section are actually part of this rule set. Altogether about twenty different conversion routines are used on phrase trees and another twenty on dependency trees.

In parallel with the development of the translation rules, the resulting dependency trees were revised by human annotators familiar with the annotation guidelines for our dependency grammar. The object was to produce dependency trees that conform to our annotation guidelines while adhering to the structure chosen by NEGRA's annotators as much as possible. In the first 3,000 automatically transformed trees, about one dependency edge in 100 had to be hand-corrected[1]. This accuracy is higher than many reports of inter-annotator agreement because inherent ambiguity has already been resolved by the NEGRA annotators. It certainly suggests that the practice of automatic translation is acceptable for practical purposes.

When applying our existing constraint grammar to the converted dependency trees, all disagreements between the two language models are automatically flagged as constraint violations. Each such case points either to an error in the translation rules, insufficient coverage in the constraint grammar, or to a mis-annotation in the original tree.

Various types of constructions were found in the NEGRA treebank that were not covered by our model of German. The majority of them have meanwhile been incorporated into the German dependency grammar. Conversely, many cases of inconsistently used edge and node tags were found in the original NEGRA corpus. While many of these correspond to actual disagreements in modelling (often in connection with the complex POS tag set of German), there are also many obviously wrong or inconsistent assignments in the NEGRA treebank. Altogether about 400 such changes were necessary in the first 3.000 sentences of the NEGRA corpus, ranging from avoidable agreement errors to phrase node subordinations that are in clear violation of NEGRA's own guidelines, and easily corrected.

5. Related work

A number of approaches to (semi-)automatically transform phrase-structure annotations to dependency trees and vice versa has been developed, most of them for English corpus data. (Eisner, 1996) e.g. uses dependency structures derived from the Penn Treebank to train a corresponding stochastic parser, while others, e.g. (Collins, 1999), following Lin's original proposal, transform both the gold standard and the parsing result into dependency space to carry out a linguistically more motivated comparison there. More recently this approach was applied to evaluate a partial parser for German on TIGER-Treebank data (Kübler and Telljohann, 2002) and it was argued that a dependency-based evaluation could considerably increase the validity also for partial parsing results.

Usually, bare-bone dependencies are used to make the desired comparisons. Even if the transformation procedure of (Kübler and Telljohann, 2002) has been designed to also obtain some edge labels, this information is apparently ignored when measuring the quality of parsing results. As long as only attachment problems are considered the transformation can be based solely on a head percolation table (Magerman, 1994), which contains the necessary information to uniquely identify the head of each phrase.

Few transformation procedures consider edge labels, which are useful to distinguish different kinds of dependency relations. (Xia et al., 2000) use an additional argument and tagset table to also distinguish between arguments and adjuncts in English sentences. (Kübler and Telljohann, 2002) are able to determine edge labels from an even richer set of categories, although this is done only for dependencies from the finite verb. Using a fully elaborate set of analytical functions, the transformation procedures for deriving Praguian tectogrammatical annotations from PTB-style phrase structure trees are most similar to the approach described in this paper. Dependency relations are assigned by a procedure taking into account the POS of the dependent node and the sequence of all its ancestors in the phrase tree as well as the POS and other lexical information of its parent in the dependency tree (Žabokrtský and Smrž, 2003). (Žabokrtský and Kučerová, 2002) report an attach-

[1]This figure drops to 98% if the dependency labels are also considered. This is in part due to information missing in the NEGRA corpus, e.g. our grammar distinguishes prepositional objects from modifying prepositionals, while NEGRA annotates them indifferently as 'MO'.

ment accuracy of 94% and a label accuracy of 82% when comparing the transformation results for English PTB annotations against manually corrected ones, a quality level which is still considerably lower than the one achieved with the DEPSY tool.

Finally, there are a few cases of programs which transform in the opposite direction. They are needed if the treebank data are given in dependency format as this is the case for the Prague Dependency Treebank. (Collins et al., 1999) produce phrase structure descriptions from dependency trees, where the derived trees have been designed to be as flat as possible. The transformed data has been used to train and evaluate a stochastic parser for Czech. (Xia and Palmer, 2001) have shown that the algorithm used by (Collins et al., 1999) is actually a special case of a more general one, using also tree-bank specific information such as the types of the arguments or modifiers a head can take. Using the extended model they achieved 88% precision and 86% recall when evaluating a "round-trip" transformation on the Penn Treebank.

6. Conclusions

We presented DEPSY, a tool for transforming phrase structure annotations to dependency trees, which can be adapted to different treebank formats and annotation guidelines. The tool has been designed

- to be used for a large scale evaluation of a dependency-based parser for German on currently available corpus data,
- to facilitate parser comparison across the two basic model classes for structural descriptions, and
- to contribute to an emerging standard of parser evaluation for German sentences as it already exists for English with the widely accepted Penn Treebank setting.

Furthermore, we intend to enrich the NEGRA corpus with detailed morphological information beyond POS tags, which is currently available for only 16% of the NEGRA sentences. With the syntax structure known, it is straightforward to let our parser compute the most fitting morphological variant for each word and this additional information can then be transferred back to the NEGRA format.

DEPSY is freely available and can be downloaded from `http://nats-www.informatik.uni-hamburg.de/download`.

7. References

Beil, Franz, Detlef Prescher, Helmut Schmid, and Sabine Schulte im Walde, 2002. Evaluation of the Gramotron parser for German. In *Proc. LREC Workshop: Beyond PARSEVAL, May 29-31*. Las Palmas, Gran Canaria.

Bouma, Gosse and Geert Kloosterman, 2002. Querying dependency treebanks in XML. In *Proc. 3rd Int. Conf. on Language Resources and Evaluation, LREC-2002*. Las Palmas, Gran Canaria.

Briscoe, Ted, John Carroll, and Antonio Sanfilippo, 1998. Parser evaluation: A survey and a new proposal. In *Proc. 1st Int. Conf. on Language Resources and Evaluation, LREC-1998*. Granada, Spain.

Collins, Michael, 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania, Philadephia, PA.

Collins, Michael, Jan Hajič, Lance Ramshaw, and Christoph Tillmann, 1999. A statistical parser for czech. In *37th Annual Meeting of the Association for Computational Linguistics, ACL-1999*. College Park, Maryland.

Daum, Michael, Kilian Foth, and Wolfgang Menzel, 2003. Constraint based integration of deep and shallow parsing techniques. In *Proc. 11th Conference of the European Chapter of the ACL*. Budapest, Hungary.

Eisner, Jason M., 1996. An empirical evaluation of probability models for dependency grammars. Technical Report IRCS-96-11.

Kübler, Sandra and Heike Telljohann, 2002. Towards a dependency-oriented evaluation for partial parsing. In *Beyond PARSEVAL, Workshop on LREC-2002*. Las Palmas, Gran Canaria.

Lin, Dekang, 1995. A dependency-based method for evaluating broad-coverage parsers. In *Proceedings of IJCAI*. Montreal, Canada.

Magerman, David M., 1994. *Natural Language Parsing as Statistical Pattern Recognition*. Ph.D. thesis, Stanford University.

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz, 1994. Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Schröder, Ingo, 2002. *A Framework for Gradation in Natural Language Analysis Using Constraint Optimization*. Ph.D. thesis, Universität Hamburg, Department of Computer Science.

Schröder, Ingo, Wolfgang Menzel, Kilian Foth, and Michael Schulz, 2000. Modeling dependency grammar with restricted constraints. *Traitement Automatique des Langues (T.A.L.)*, 41(1):113–144.

Skut, Wojciech, Thorsten Brants, Brigitte Krenn, and Hans Uszkoreit, 1997. Annotating unrestricted german text. In *Proc. 6. Fachtagung der Sektion Computerlinguistik der Deutschen Gesellschaft für Sprachwissenschaft*. Heidelberg, Germany.

Žabokrtský, Zdeněk and Ivona Kučerová, 2002. Transforming Penn Treebank phrase trees into (Praguian) tectogrammatical dependency trees. *Prague Bulletin of Mathematical Linguistics*, 78:77–94.

Žabokrtský, Zdeněk and Otakar Smrž, 2003. Arabic syntactic trees: From constituency to dependency. In *11th Conf. of the European Chapter of the ACL, EACL-2003*. Budapest, Hungary.

Xia, Fei and Martha Palmer, 2001. Converting dependency structures to phrase structures. In *Proc. Int. Conf. on Human Language Technology, HLT-2001*. San Diego, CA.

Xia, Fei, Martha Palmer, and Aravind Joshi, 2000. A uniform method of grammar extraction and its application. In *Proc. Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP/VLC-2000*. Hong Kong.