Comparing Dependency and Constituent Syntax for Frame-semantic Analysis

Richard Johansson, Pierre Nugues

Lund University
{richard, pierre}@cs.lth.se

Abstract

We address the question of which syntactic representation is best suited for role-semantic analysis of English in the FrameNet paradigm. We compare systems based on dependencies and constituents, and a dependency syntax with a rich set of grammatical functions with one with a smaller set. Our experiments show that dependency-based and constituent-based analyzers give roughly equivalent performance, and that a richer set of functions has a positive influence on argument classification for verbs.

1. Introduction

The role-semantic paradigm (Gildea and Jurafsky (2002), *inter alia*) is a prominent model in automatic semantic analysis with a wide range of proposed applications. With a few exceptions, role-based semantic analysis relies crucially (Gildea and Palmer, 2002; Punyakanok et al., 2005) on some sort of syntactic representation as input. This enables the analyzer to extract syntactic features that are used by statistical classifiers. The connection between syntax and semantics has also been noted in annotation projects of semantic treebanks; for instance, the SALSA project (Burchardt et al., 2006) and the Prague Dependency Treebank (Hajič, 1998) have annotated semantic structures on top of syntactic treebanks.

The CoNLL 2004 and 2005 shared tasks (Carreras and Màrquez, 2005) were the first events that conducted an impartial evaluation of role-semantic labelers on the PropBank corpus. More recently, SemEval organized a similar evaluation (Baker, 2007) using the FrameNet corpus. While nearly all participants in the CoNLL shared tasks used a constituent representation that underlies the PropBank annotation, the best-performing system (Johansson and Nugues, 2007b) of SemEval 2007 used dependency graphs.

These evaluations are not directly comparable however. They use different corpora, annotation, and training methods and from these results, it is difficult to conclude what is the optimal representation to use in the parsing step. Some grammatical features used in constituents and dependencies may be directly equivalent. Conversely, some other features are tied to a specific representation. In addition, the contribution of the features and their behavior as a function of the training set and its size would still need more exploration.

This paper addresses the question of how the syntactic representation influences the performance of automatic semantic analysis in the paradigm of Frame Semantics (Fillmore, 1976; Baker et al., 1998). To study the impact of syntactic representations, we performed a set of experiments in which we compare the performance of constituent-based and dependency-based semantic analyzers on the three main subtasks of FrameNet-based role-semantic analysis.

2. Automatic Frame-Semantic Analysis

The task of frame-semantic analysis is usually divided into three main subtasks: detection and disambiguation of target words, detection of semantic arguments, and finally classification of arguments (see Figure 1).

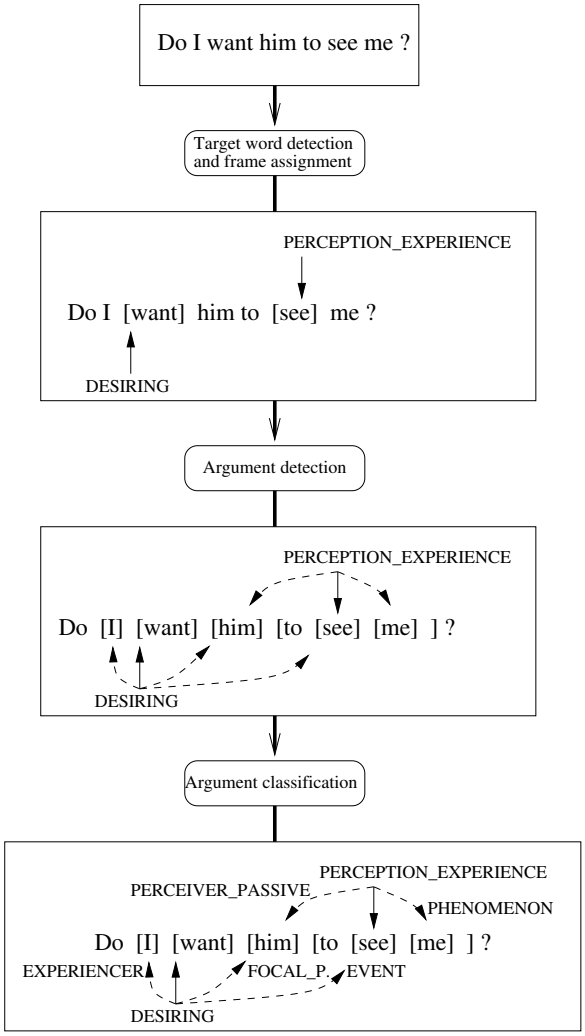

Figure 1: The stages in the frame-semantic structure extraction process.

Although it is generally agreed that the best performance is obtained when the tasks are solved jointly, it is easiest from an engineering point of view, and computationally less expensive, to treat them as independent tasks that are solved sequentially.

For both the constituent-based and the dependency-based

semantic analyzer, we implemented the three subtasks as statistical classifiers using support vector machines; the first classifier assigns a frame for a given target word, the second decides whether a given node in a constituent or dependency parse tree represents an argument for a given predicate, and the final one assigns a semantic role to a node that has been identified as an argument by the previous step. The second step also makes use of a set of filtering rules to reduce the number of potential arguments (Xue and Palmer, 2004). Table 1 shows the features used by the classifiers. The features used by the constituent-based and the dependency-based classifiers are indicated by C and D, respectively.

Features	Target disambiguation	Argument identification	Argument classification
FRAMES	C,D		
TARGETLEMMA	C,D	C,D	C,D
CHILDWORDSET	C,D		
PARENTWORD	C,D		
FES		C,D	C,D
TARGETPOS		C,D	C,D
VOICE		C,D	C,D
POSITION		C,D	C,D
ARGWORD/POS		C,D	C,D
LEFTWORD/POS		C,D	C,D
RIGHTWORD/POS		C,D	C,D
C-SUBCAT	C	C	C
C-PATH		C	C
PHRASETYPE		C	C
GOVCAT		C	C
D-SUBCAT	D	D	D
D-PATH		D	D
FUNCTION			D
CHILDDEPSET	D	D	D
CHILDWORDDEPSET	D		

Table 1: Features used by the classifiers.

The following three subsection describes the features used by the classifiers. All examples are given with respect to Figure 2.

2.1. Common Features

The following features are used by both the constituent-based and the dependency-based semantic analyzers. Head-finding rules (Johansson and Nugues, 2007a) were applied when heads of constituents were needed.

FRAMES. The set of frames listed in FrameNet for a lemma. For instance, for the verb *want*, FrameNet lists DESIRING and POSSESSION.

TARGETLEMMA. The lemma of the target word itself, e.g. *want*.

CHILDWORDSET. The set of dependent head words of the target word. For *see*, this set is { *to*, *me* }.

PARENTWORD. The word of the parent word of the target. For *see* in the example, this is *want*.

FES. For a given frame, the set of available frame elements listed in FrameNet. For instance, for *see* in the PERCEPTION_EXPERIENCE frame, we have 12 frame elements: DEGREE, PERCEIVER_PASSIVE, PHENOMENON, ...

TARGETPOS. Part-of-speech tag for the target word.

VOICE. For verbs, this feature is Active or Passive. For other types of words, it is not defined.

POSITION. Position of the head word of the argument with respect to the target word: Before, After, or On.

HEADWORD and HEADPOS. Word and part-of-speech tag of the argument.

LEFTWORD and LEFTPOS. Word and part-of-speech tag of the leftmost dependent of the argument head.

RIGHTWORD and RIGHTPOS. Word and part-of-speech tag of the rightmost dependent of the argument head.

2.2. Features Used by the Constituent-based Analyzer

C-SUBCAT. Subcategorization frame: corresponds to the phrase-structure rule used to expand the phrase around the target. For *want* in the example, this feature is `VP→VB S`.

C-PATH. A string representation of the path through the constituent tree from the target word to the argument constituent. For instance, the path from *want* to *I* is `↑VP-↑SQ-↓NP`.

PHRASETYPE. Phrase type of the argument constituent, e.g. `NP` for *him*.

GOVCAT. Governing category: this feature is either `S` or `VP`, and is found by starting at the argument constituent and moving upwards until either a `VP` or a sentence node (`S`, `SINV`, or `SQ`) is found. For instance, for *him*, this feature is `S`, while for *me*, it is `VP`. This can be thought of as a very primitive way of distinguishing subjects and objects.

2.3. Features Used by the Dependency-based Analyzer

D-SUBCAT. Subcategorization frame: the grammatical functions of the dependents concatenated. For *want*, this feature is `OBJ+OPRD`.

D-PATH. A string representation of the path through the dependency tree from the target node to the argument node. Moving upwards through verb chains is not counted in this path string. In the example, the path from *want* to *I* is `↓SBJ`.

FUNCTION. The grammatical function of the argument node. For direct dependents of the target, this feature is identical to the D-PATH.

CHILDDEPSET. The set of grammatical functions of the direct dependents of the target node. For instance, for *want*, this set is { `OBJ`, `OPRD` }.

CHILDWORDDEPSET. The set of word/function pairs of the dependents of the target. For instance, for *want*, this set is { him-`OBJ`, see-`OPRD` }.

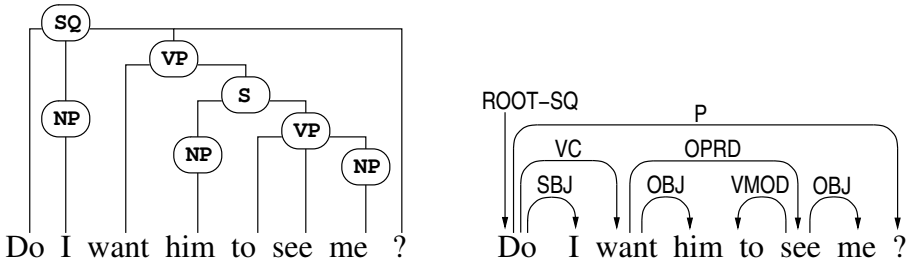

Figure 2: Constituent and dependency trees for the example sentence.

3. Experiments

The purpose of the experiments is twofold: first, to investigate whether dependency-based semantic analyzers are competitive with constituent-based analyzers; secondly, to measure the influence of the richness of the set of grammatical functions. In addition, we would like to measure the influence of formalisms rather than of parsing performance, so we used two parsers of each type.

To train and evaluate the dependency-based semantic analyzers, we parsed the training and test corpora using two freely available parsers: MALTPARSER (Nivre et al., 2007) and MSTPARSER (McDonald and Pereira, 2006). They were both trained on a dependency treebank that had been automatically converted from the Penn Treebank using the LTH constituent-to-dependency conversion tool (Johansson and Nugues, 2007a), achieving a labeled accuracy of 87.4% and 86.9% on section 23 of the WSJ part of the Treebank, respectively. In addition, we trained a model for MALTPARSER using PENN2MALT[1] which gives a dependency representation with a smaller set of grammatical functions. This model achieves a labeled accuracy of 90.3% on WSJ section 23.

For the constituent-based analyzers, we used the popular Collins' (1997) and Charniak's (2000) parsers. Although grammatical functions (subject, locative, temporal, ...) are available in the Treebank, they are not available in the output of these parsers. The published labeled precision and recall figures for these parsers are 88.1/87.5 and 89.5/89.6, respectively. Collins' parser comes with three different parsing models; in the experiments, we report the result for the best-performing of these.

This gives us in total five semantic analyzers that we studied for each experiment: MALTPARSER and MSTPARSER with LTH-style dependencies, MALTPARSER with PENN2MALT dependencies, and Collins' and Charniak's constituent parsers. In the experiments, we studied only target words that were adjectives, adverbs, noun, and verbs, since annotated data are more reliable for these word classes. All tests were run on the test data from the SemEval-2007 task on Frame-semantic Structure Extraction (Baker, 2007). The test corpus consists of 120 sentences and contains 970 target words and 1,663 semantic arguments, not counting null-instantiated arguments.

[1] http://w3.msi.vxu.se/∼nivre/research/Penn2Malt.html

3.1. Target Word Detection and Frame Disambiguation

Table 2 shows the precision, recall, and F1 measures for target word detection and disambiguation. In addition, the last column shows the disambiguation accuracy when the target word is given. When a number appears with an asterisk, this denotes that the difference between this figure and the best figure has at least 95% statistical significance according to a McNemar test.

Parser	P	R	F1	Accuracy
Malt/P2M	74.1	68.8	71.4	85.7
Malt/LTH	73.1	67.8	70.3	84.5*
MST/LTH	73.8	68.4	71.0	85.1
Charniak	74.4	70.1	72.2	85.4
Collins	73.7	68.5	71.1	86.6

Table 2: Target word detection/disambiguation performance.

Interestingly, the performance on this task seems to be negatively affected by the rich set of grammatical functions – the two parsers that use the LTH dependency format score lower than the dependency parser that uses the PENN2MALT format and the two constituent parsers.
Since the dependency-based disambiguation classifiers use more features than their constituent-based counterparts, we believe that this difference may be a case of the "curse of dimensionality" – the large number of features makes the learning curve rise slowly. We intend to carry out a series of feature engineering experiments to investigate this more thoroughly.

3.2. Semantic Argument Detection

In the next experiment, we investigated the performance of argument detection when target words and frames were given. An argument was counted as correctly detected if its bracketing coincided with the bracketing in the gold standard, disregarding punctuation. Table 3 shows the precision, recall, and F1 measures. The table gives results for all targets and for verb targets.
The most striking discrepancy in the table is the low performance of the semantic analyzer based on MSTPARSER; the differences are much larger than the difference in parsing accuracy on the WSJ. A possible reason for this difference may be that MSTPARSER reportedly performs slightly worse than MALTPARSER on short-distance links, which possibly would include most arguments.

	All targets			Verb targets		
Parser	P	R	F1	P	R	F1
Malt/P2M	62.1	52.0	56.6	64.5	57.5	60.8
Malt/LTH	65.6	53.1	58.7	70.2	58.8	64.0
MST/LTH	57.6	51.2	54.2	58.4	54.9	56.6
Charniak	65.8	51.7	57.9	73.9	61.7	67.3
Collins	63.4	53.0	57.7	73.5	61.1	66.7

Table 3: Semantic argument detection performance.

The other two dependency-based analyzers perform well, especially the model using LTH-style grammatical functions, which outperforms the constituent-based analyzers by 0.8 points for argument detection over all targets.
For verbs, on the other hand, the constituent-based analyzers outperform all the dependency-based ones by a wide margin. Inspection of the output data indicates that a part of the low figures for the dependency-based systems may be explained by ambiguity introduced by coordination and raising.

3.3. Semantic Argument Classification

In the final experiment, we gave the system as input the target words and the corresponding frames, and the argument bracketings. Table 4 shows the argument classification accuracy for all targets and for verb targets.
Here, the analyzers based on LTH-style dependencies outperform the constituent-based analyzers. This difference is even more prominent for verbs; we believe that this is because the LTH dependency graphs use 39 different labels for verb arguments and adjuncts, while the PENN2MALT graph only use five. For non-verb modifiers on the other hand, the LTH and PENN2MALT dependency styles use the same labels.

Parser	All targets	Verb targets
Malt/P2M	69.5	70.5*
Malt/LTH	69.9	74.2
MST/LTH	70.4	73.6
Charniak	67.8*	69.8*
Collins	68.9	72.5

Table 4: Semantic role classification accuracy.

To further assess the influence of grammatical function on the classification accuracy, we performed a feature engineering study. For the dependency-based analyzers, the information about the grammatical function is encoded in the D-PATH and FUNCTION features. Table 5 shows the classification accuracy varies according to the feature set. It is clear that at least one of D-PATH and FUNCTION is necessary for accurate classification. Interestingly, it seems that the best performance is achieved when D-PATH is left out. This also reduces the classifier complexity since FUNCTION has a much smaller range than D-PATH.
We performed a similar experiment for the constituent case (Table 6); here, the information about grammatical function is encoded in the C-PATH and GOVCAT features. Here, the result suggests that both features are necessary for best accuracy – the C-PATH feature should not be removed, which gives a more complex classifier since the number of possible values for C-PATH is very large.

Feature set			
D-PATH	FUNCTION	All targets	Verb targets
+	+	69.9	74.2
-	+	70.0	75.3
+	-	69.4	74.8
-	-	68.5*	72.3*

Table 5: Dependency-based role classification accuracy by feature set.

Feature set			
C-PATH	GOVCAT	All targets	Verb targets
+	+	67.8	69.8
-	+	66.9	70.1
+	-	66.9	69.3
-	-	66.8	68.5*

Table 6: Constituent-based role classification accuracy by feature set.

4. Perspectives

For English, there exist both constituent and dependency parsers, and this study has shown that frame-semantic analyzers can use either representation. However, many languages have only one type of parser as for Danish or Czech where annotated corpora only use dependencies. We hope these experiments will help clarify the design of semantic parsers by itemizing the available features and outlining their contribution.
The most significant difference seems to be whether the output syntactic structure contains information about grammatical functions or not. In this study, we compared a dependency grammar that had a rich set of functions (39 labels) for verb dependents with one that had a small set (five labels), and showed that the richer set leads to a significant improvement in argument classification accuracy for verbs. For noun and adjective modifiers on the other hand, there was no difference in the set of grammatical functions, and consequently no difference in classification performance.

5. References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of COLING-ACL*.

Collin Baker. 2007. SemEval task 19: Frame semantic structure extraction. In *Proceedings of SemEval*.

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of LREC*.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of CoNLL-2005*.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL*.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of ACL-EACL*.

Charles J. Fillmore. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language*, 280:20–32.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Daniel Gildea and Martha Palmer. 2002. The necessity of syntactic parsing for predicate argument recognition. In *Proceedings of ACL*.

Jan Hajič. 1998. Building a syntactically annotated corpus: The Prague Dependency Treebank. In *Issues of Valency and Meaning*, pages 106–132.

Richard Johansson and Pierre Nugues. 2007a. Extended constituent-to-dependency conversion for English. In *Proceedings of NODALIDA*.

Richard Johansson and Pierre Nugues. 2007b. Semantic structure extraction using nonprojective dependency tress. In *Proceedings of SemEval-2007*.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2005. The necessity of syntactic parsing for semantic role labeling. In *Proceedings of IJCAI*.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP*.