

Enriching Frame Semantic Resources with Dependency Graphs

Hagen Fürstenu

Dept. of Computational Linguistics, Saarland University
Saarbrücken, Germany
hagenf@coli.uni-saarland.de

Abstract

We propose two general and robust methods for enriching resources annotated in the Frame Semantic paradigm with syntactic dependency graphs, which can provide useful additional information for applications such as semantic role labeling methods. One method incorporates information of a dependency parser, while the other one assumes the resource to be based on a treebank and uses dependency graphs converted from phrase structure trees. Coverage and accuracy of the methods are evaluated on the English FrameNet and German SALSA corpora. It is shown that large proportions of those resources can be accurately enriched by mapping their annotations onto dependency graphs. Failures to do so are found to be largely due to parser errors and can therefore be seen as an indicator of incorrect parses, which helps to improve parse selection. The remaining failures are analyzed and an outlook on ways of improving the results by adaptation to specific resources is given.

1. Introduction

Recent years have seen a growing interest in resources annotated within the Frame Semantics paradigm (Fillmore, 1982), e.g. the corpora created in the English FrameNet (Baker et al., 1998) or German SALSA (Burchardt et al., 2006) projects. Information about semantic roles contained in such resources has been employed to train semantic role labeling methods, which are then capable of providing semantic analyses for unseen sentences, a process that has been called “semantic parsing”. Applications of such analyses are manifold, e.g. in question answering (Shen and Lapata, 2007) or information retrieval (Moschitti et al., 2003). Semantic role labeling methods or similar applications, however, usually require syntactic analyses of the sentences in their training corpus as well, which can be either included in the resource, e.g. in the form of a treebank, or obtained by syntactic parsing. While most methods proposed so far use phrase structure trees as syntactic formalism, dependency graphs are increasingly recognized as a more natural syntactic representation for semantic role labeling, promising improved performance of the latter (cf. e.g. (Hacioglu, 2004) or the CoNLL 2008 Shared Task on “Joint Parsing of Syntactic and Semantic Dependencies”¹). Advantages of dependency graphs include the more immediate representation of predicate-argument structures and the possibility of abstraction from surface phenomena irrelevant to Frame Semantics, e.g. auxiliary verbs or control structures. The existing Frame Semantic resources, however, either lack deep syntactic structure altogether (like FrameNet) or are annotated on phrase structure trees rather than dependency graphs (like SALSA).

In this paper we present and discuss two general methods for enriching Frame Semantic resources with dependency graphs. We show that the methods can robustly and accurately enrich large proportions of such resources with dependency graphs. Section 2. introduces two Frame Semantic resources which we tested our approach on. The two methods for enriching them are described in Section 3. In

Section 4. we give evaluation results of several test settings, showing the robustness of the methods and evaluating the resulting resources. Section 5. concludes with a summary of the results.

2. Data

2.1. FrameNet Corpus

The FrameNet corpus contains more than 135,000 frame annotations on example sentences drawn mostly from the British National Corpus. We only consider annotations with verbal frame evoking elements, since the advantage of dependency structures is more obvious for these and semantic role labelling methods as well as other annotation projects like SALSA have concentrated on them. There are 61,792 such annotations in the FrameNet data. Each annotation contains a frame name, e.g. *Motion_noise* and a markup of the textual extent of the frame evoking element (FEE) and each frame element (role), e.g.

[*Theme* They] [*FEE* clattered] [*Goal* into the hallway].

There is a layer of syntactic annotation in the FrameNet corpus, but it is shallow and incomplete and we therefore only consider the spans of the annotated FEE and roles of each sentence, which are given as sets of character positions (e.g. {15, . . . , 30} for the role *Goal* in the example).

2.2. SALSA Corpus

In the SALSA corpus there are around 20,000 frame annotations on sentences of the TIGER treebank. We discard annotations with non-verbal FEEs as targets or underspecified frame assignments (where one verb is annotated with multiple frames), so that 18,086 annotations remain. A SALSA annotation consists of the TIGER phrase structure tree together with information on which tree nodes make up the FEE and the roles. As an example, Figure 1 shows the annotation of a sentence with the frame *Awareness*. The FEE and the role *Cognizer* are made up of terminal nodes (word tokens), while the role *Content* is associated with a non-terminal *S* node of the tree.

¹<http://www.yr-bcn.es/con112008/>

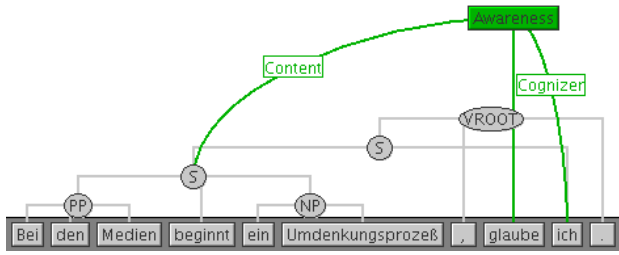


Figure 1: Example SALSA annotation (*I believe a process of rethinking is starting within the media.*)

For comparison with the FrameNet corpus, the SALSA corpus can easily be stripped of its underlying treebank structure, so that the data have the same form as in the FrameNet corpus:

[*Content* Bei den Medien beginnt ein Umdenkungsprozess],
 [*FEE* glaube] [*Cognizer* ich].

3. Methods

A dependency analysis of a sentence can be represented as a directed graph with nodes labeled by word token lemmata and edges labeled by grammatical relations. (Lemmatization may lead to graph nodes representing more than one token or one token being represented by more than one graph node, e.g. in the cases of separable affixes or compounds.) Depending on the underlying syntactic theory such dependency graphs may exhibit undirected cycles (i.e. not be trees) or even directed cycles (i.e. not be directed acyclic graphs, DAGs). Figure 2 shows a possible analysis of the sentence “*He wanted to eat some fried shrimps.*” illustrating both of these cases. Here *he* is analysed as (deep) subject of both *want* and *eat*, creating an undirected cycle. A directed cycle is formed because *fry* is analysed as a modifying dependent of *shrimp* while at the same time *shrimp* is the object of *fry*.

All graph nodes may carry additional information about the word tokens they represent, such as POS tags or morphological analyses.

When enriching a Frame Semantically annotated corpus with dependency graphs, our goal is to produce semanti-

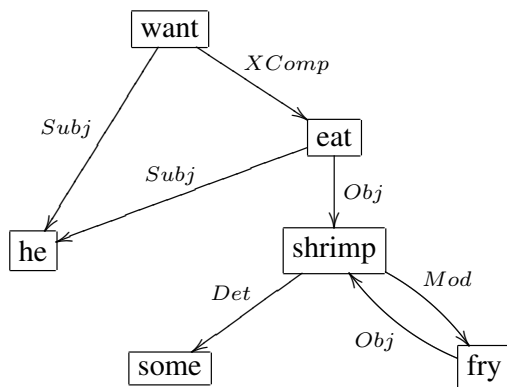


Figure 2: Dependency graph with directed and undirected cycles

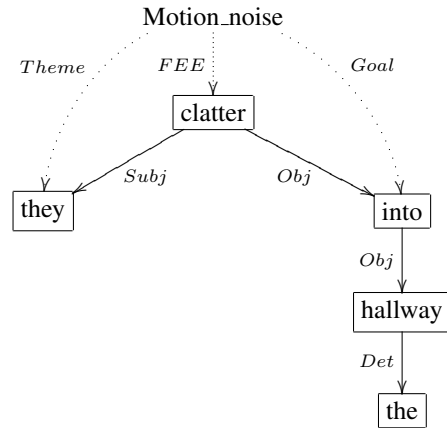


Figure 3: Semantically annotated dependency graph

FEE and role spans:	
S_{FEE}	$= \{5, \dots, 13\}$
S_{Theme}	$= \{0, \dots, 3\}$
S_{Goal}	$= \{15, \dots, 30\}$
node spans:	
$S_{clatter}$	$= \{5, \dots, 13\}$
S_{they}	$= \{0, \dots, 3\}$
S_{into}	$= \{15, \dots, 18\}$
$S_{hallway}$	$= \{24, \dots, 30\}$
S_{the}	$= \{20, \dots, 22\}$
subgraph spans:	
$S_{clatter}$	$= \{0, \dots, 30\} \setminus \{4, 14, 19, 23\}$
S_{they}	$= \{0, \dots, 3\}$
S_{into}	$= \{15, \dots, 30\} \setminus \{19, 23\}$
$S_{hallway}$	$= \{20, \dots, 30\} \setminus \{23\}$
S_{the}	$= \{20, \dots, 22\}$

Figure 4: Spans considered by the Span Comparison method for the example of Figure 3

cally annotated dependency graphs by merging the Frame Semantic and dependency analyses. In such an annotated dependency graph, the FEE and each role of a frame is associated with certain graph nodes. Figure 3 shows this for the example sentence of Section 2.1.

3.1. Mapping by Span Comparison

For Frame Semantic corpus resources which do not include syntactic analyses, the sentences first have to be parsed into dependency graphs. We assume that each graph node is marked up with the spans of the word token or tokens it represents and call this set of indices the *node span*. The Frame Semantic annotation in turn provides us with the spans of the FEE and each role, which we want to call the *FEE span* and the *role spans* respectively. These spans are shown in the first two blocks of Figure 4.

Since we expect the verbal FEE of a frame to correspond to a single graph node, we try to identify a node whose node span is compatible with the FEE span.

If s_n is the node span of a node n of the graph G and s_{FEE} is the FEE span, we define the set of compatible nodes as

$$N_{\text{FEE}} := \left\{ n \in G \mid s_n = s_{\text{FEE}} \cap \bigcup_{n \in G} s_n \right\}$$

The intersection with the union of all node spans (i.e. the total graph span) allows for parts of the FEE to be dropped, provided they do not occur in the dependency graph at all. This is e.g. the case for separable verb prefixes in German, which may disappear in lemmatization. If

$$|N_{\text{FEE}}| = 1$$

i.e. a unique compatible node has been found, we associate this node with the FEE, otherwise the method fails to map the semantic annotation onto the dependency graph.

Having identified the FEE with a graph node, we proceed to associate the roles of the frame annotation with graph nodes. For this a different approach has to be employed, because a role is normally expected to consist of a subgraph of the dependency graph. Therefore, we first determine the *subgraph span* of each node, which in general is the union of the node spans of the node itself and all graph nodes directly or indirectly dominated by it.

An exception from this definition has to be made in the case of nodes with multiple in-edges. Consider e.g. the node *he* in Figure 2. In general it is useful to know that it constitutes a deep subject of *eat*. A frame annotation with *want* as FEE, however, might look like this:

[*Experiencer* He] [*FEE* wanted]
[*Event* to eat some fried shrimps].

In order to identify *eat* as the graph node corresponding to the *Event* role, we therefore have to exclude the node *he* from the subgraph span of *eat*. Formally we define the subgraph span S_n as

$$S_n := \bigcup \{ s_{n'} \mid (n' \leq n) \wedge (\forall p \in \text{pred}(n') : p \leq n) \}$$

where \leq denotes the reflexive relation of (direct or indirect) dominance and $\text{pred}(n')$ is the set of direct predecessors of n' . What this means is that we exclude descendants of a node from its subgraph span if and only if there is an edge to them from “outside the subgraph”. Cyclic structures entirely contained within the subgraph on the other hand, such as the directed cycle resulting in two in-edges at the node *shrimp*, do not cause the relevant nodes to be excluded. The subgraph spans for our example are shown in the third block of Figure 4. (For this simple example no corrections due to multiple in-edges have to be made.)

We define the set of compatible graph nodes for each role r analogous to N_{FEE} , with the only exception that we now compare the role span s_r to subgraph spans S_n instead of node spans s_n :

$$N_r := \left\{ n \in G \mid S_n = s_r \cap \bigcup_{n \in G} s_n \right\}$$

Intersection with the total graph span is even more vital in this case, as it allows us to ignore mismatches due to

whitespace or punctuation included in role spans, but not represented in the dependency graph. If

$$|N_r| = 1$$

for all roles r , then we associate all roles with their unique compatible graph nodes, otherwise the mapping fails. Applying the method to our example sentence results in

$N_{\text{FEE}} = \{\text{clatter}\}$
 $N_{\text{Theme}} = \{\text{they}\}$
 $N_{\text{Goal}} = \{\text{into}\}$

which is the expected outcome of Figure 3.

3.2. Mapping by Node Correspondence

In the case of a resource like the SALSAs corpus, where semantic annotation is anchored to nodes of phrase structure trees, there is an easier way of generating annotated dependency graphs. For this the trees have to be converted into dependency graphs which for each graph node retain information about which tree nodes it corresponds to. Then a direct projection of the semantic annotations from trees to graphs is possible. A schematic example of this is shown in Figure 5. The phrase structure tree on top is annotated with a semantic frame (dotted arrows). This is uniquely mapped onto the dependency graph below by means of node correspondences (dashed arrows).

If mapping of the FEE and each role is possible and unambiguous, i.e. each role bearing tree node corresponds to exactly one graph node, then the projection is successful. This is not always the case though and the method may fail if the correspondences are ambiguous or erroneous.

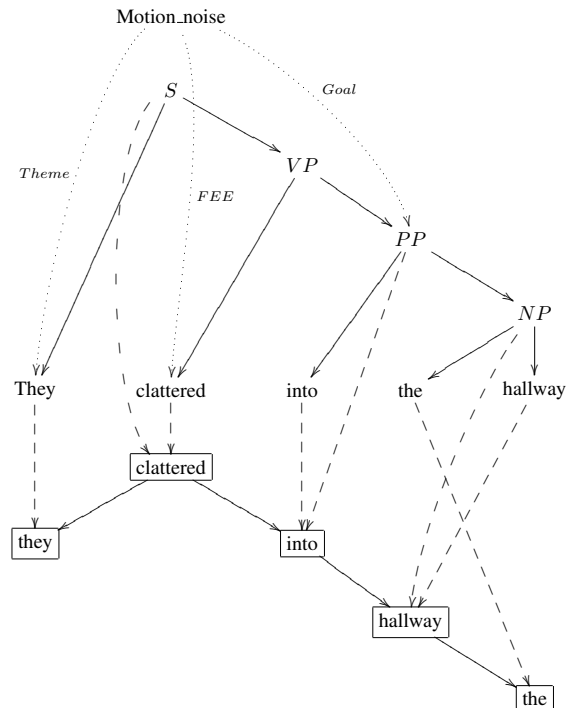


Figure 5: Corresponding annotated phrase structure tree and dependency graph

Dependency Graphs	Mapping method	Mappable	Correct (of those mappable)	Parser Errors (of those not mappable)
Best LFG parse	Span Comparison	49.3%	76% ($\pm 9\%$)	82% ($\pm 8\%$)
Best RASP parse	Span Comparison	44.4%	78% ($\pm 9\%$)	76% ($\pm 9\%$)
20 best LFG parses	Span Comparison	61.1%	77% ($\pm 9\%$)	83% ($\pm 8\%$)
20 best RASP parses	Span Comparison	65.1%	78% ($\pm 9\%$)	78% ($\pm 9\%$)

Table 1: Results for the English FrameNet sample

4. Experiments and Results

We applied the two methods described in Section 3. to enrich the FrameNet and SALSA corpora with dependency graphs. The Span Comparison method of Section 3.1., not relying on syntax information in the resource, was applied to both corpora, showing that it works independent of corpus and language. Results on the FrameNet corpus are discussed in Section 4.1., those on the SALSA corpus (stripped of its treebank information) in Section 4.2.1. The Node Correspondence method of Section 3.2., using annotated phrase structure trees, was applied to the SALSA corpus in two different approaches, detailed in Sections 4.2.2. and 4.2.3.

For the Span Comparison method, which relies on parser output, we considered both the best ranking and the 20 best ranking analyses provided by the parsers. In the latter case, the method was applied to the readings successively until it could annotate one or was found to fail on all of them.

For each examined combination of mapping and dependency graph generation method the correctness of the resulting enriched resource was assessed by manually evaluating a random sample of 100 annotated dependency graphs. Since complete correctness of a dependency graph is a very strict measure when parsing long and complex sentences, an evaluation oriented on typical applications such as semantic role labeling methods was carried out: An annotated dependency graph was classified as correct if the paths from the node annotated as FEE to each node annotated with a role were correct as judged by a human evaluator. Only when the role-bearing node represented a preposition or a coordination, its dependents, which in this case are the semantically significant head words of the role, were also checked. Arguments and adjuncts were not distinguished, since in Frame Semantics this distinction is supposed to be captured by the status of a role (i.e. core, peripheral or extra-thematic role).

To assess the reasons for failures of the methods, a random sample of 100 dependency graphs which could not be semantically annotated was examined in each setting and the proportion of failures due to (or partly due to) parser errors was determined.

4.1. FrameNet corpus

We parsed the 61,792 sentences of the FramNet corpus featuring verbal FEEs with both the English LFG developed in the ParGram project (Riezler et al., 2003), which is a hand-crafted grammar with statistical parse ranking, and with RASP (Briscoe et al., 2006), a statistical parser producing dependency graphs. The f-structures of the LFG

were converted to dependency graphs with rewriting rules by Tracy H. King.

The results of applying the Span Comparison method to the best and 20 best parses of both the LFG and RASP are shown in Table 1, with 95% confidence intervals for the sample-based evaluation figures. The results are quite similar between the two parsers, suggesting that our method does not depend on a specific format of dependency graphs.

While slightly less than half of the FrameNet corpus could be enriched with the best ranking parses alone, these figures increase significantly if the 20 best ranking parses are considered. Since correctness does not seem to be affected, this indicates that the additional bracketing information, which is implicit in semantic role annotations in so far as it correlates with constituent boundaries, is able to improve the parse selection process considerably. This is because wrong higher ranking parses are discarded in favour of correct lower ranking ones if the annotation can be mapped to the latter but not the former.

The relatively high number of unmappable annotations was found to be largely due to parser errors, often attachment problems leading to mismatches with the subgraph spans. Since with the current state of the art in dependency parsing a certain proportion of errors cannot be avoided, our attention is on how well our method can detect them and discard the problematic sentences. Ideally the method would process all correct parses and fail on all erroneous parses. The figures in the columns “Correct” and “Parser Errors” of Table 1 can therefore be seen as measures of how close to this ideal our method performs.

4.2. SALSA corpus

4.2.1. Reparsing

For comparison with the experiments on the FrameNet corpus, we first stripped the SALSA corpus of its treebank structure and parsed the 18,086 sentences with the German LFG of the ParGram project (Dipper, 2003; Rohrer and Forst, 2006), ranking the parses with the method described in (Forst, 2007) and generating dependency graphs with rewriting rules by Martin Forst. The results of applying the Span Comparison method, shown in the first two rows of Table 2, are similar to those for the FrameNet corpus, indicating that the method is not only parser-independent, but also independent of language and corpus. The smaller relative gain obtained by considering the 20 best ranking parses (14% compared to 24% for the English LFG) might be due the more advanced parse ranking method employed.

Dependency Graphs	Mapping method	Mappable	Correct (of those mappable)	Parser Errors (of those not mappable)
Best LFG parse	Span Comparison	56.7%	81% ($\pm 8\%$)	79% ($\pm 8\%$)
20 best LFG parses	Span Comparison	64.5%	80% ($\pm 8\%$)	80% ($\pm 8\%$)
Random converted tree	Node Correspondence	97.4%	94% ($\pm 5\%$)	n/a
Most similar LFG parse	Node Correspondence	85.4%	79% ($\pm 8\%$)	n/a

Table 2: Results for the German SALSA sample

4.2.2. Converting TIGER trees

Next we converted the TIGER treebank to LFG f-structures with the method of (Forst, 2003). This conversion process is ambiguous, but since ranking of the converted f-structures is not possible with the employed models we had to choose one of the converted f-structures for each sentence at random. These f-structures were converted to dependency graphs as before and the Node Correspondence method was applied. (While this approach is similar to the one carried out in (Frank and Semecký, 2004), our method operates on the resulting dependency graphs and therefore does not have to be adapted to the particulars of the LFG f-structures.)

The results are shown in the third row of Table 2. The conversion procedure only in exceptional cases fails to preserve unique correspondences and therefore almost the whole corpus can be enriched with the converted structures. As expected, correctness is found to be significantly higher than for the parsed sentences, with only a few conversion errors.

4.2.3. Hybrid dependency graph generation

Although it seems unlikely that coverage and accuracy on the converted trees can be further improved by incorporating information from the parsed sentences, we carried out a hybrid approach of converting and reparsing the SALSA corpus for two reasons: On the one hand a parser may well provide us with useful additional information about the nodes of a dependency graph, such as better morphological analyses. On the other hand it may be desirable to enrich a Frame Semantic resource with dependency graphs of the exact format a particular parser produces. That way ap-

plications like semantic role labeling methods can straightforwardly compare sentences in the resource with parsed sentences.

We therefore evaluated an approach to enriching the SALSA corpus which uses the converted treebank information as one factor in parse selection. The idea is to compare all parses of a sentence to all readings of the ambiguous converted f-structure and choose the parse which is most similar to one of the converted f-structures. If several parses rank equally, the statistical parse ranking model is employed to choose among those. The similarity between parsed and converted structures is computed on the basis of shared f-structure facts with a script by Martin Forst, which we modified to insert tree node correspondence facts from the converted into the parsed structures. For technical limitations the sets of structures to be compared had to be restricted in cases where they were too large (a maximum of 10,000 combinations was allowed). Furthermore parses with too low similarity scores (matching less than 50% of the converted f-structure facts) were discarded.

The dependency graphs which we finally arrive at combine the advantages of both conversion and reparsing, comprising tree node correspondence information and deep analyses provided by the LFG. Figure 6 schematically shows the Node Correspondence method being applied to dependency graphs generated with the hybrid approach. The results are shown in the last row of Table 2. (Note that the Span Correspondence method does not fail because of parsing errors, so this evaluation was not meaningful, even though parsed dependency graphs were annotated.)

With a coverage of 85.4% this approach is indeed a compromise between reparsing and treebank conversion. Somewhat surprisingly, accuracy seems unchanged compared to the Span Comparison method. This can be explained as the superposition of two effects in opposite directions: On the one hand, the Node Correspondence method is more permissive and less sensitive to parser errors, resulting in lower accuracy. On the other hand, matching against converted treebank structures probably results in significantly less parser errors, countering the former effect. All in all, the hybrid approach of dependency graph generation together with the Node Correspondence method is a practical way of enriching the SALSA corpus in the case that information provided by the parser needs to be incorporated.

4.3. Error analysis

As we have seen, the majority of failures of the Span Comparison method is due to parser errors. We want to analyze some of the other failures. For the FrameNet cor-

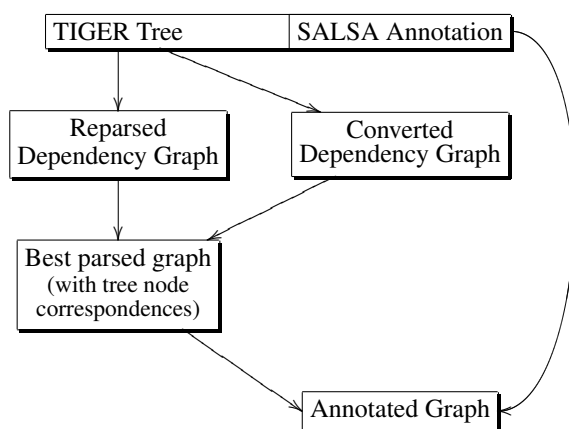


Figure 6: Schematic view of applying the Node Correspondence method in the hybrid approach

pus the three most frequent problems – together accounting for about half of those failures – are the annotation of relative pronouns, non-constituent roles and roles coinciding with the FEE. Relative pronouns in FrameNet are annotated together with their referents in roles crossing constituent boundaries, e.g. in the following annotation with the frame *Evoking*:

Use then, as you roughly plan your book,
 [*Stimulus* some locality which] [*Cognizer* for you]
 [*FEE* evokes] [*Phenomenon* a strong mood].

While the Span Comparison method could be adapted to deal with this problem (provided the relative clause was correctly identified by the parser), there are many less systematic phenomena of non-constituent role annotations, as e.g. in

[*Self_mover* The economy] continued to [*FEE* stagger]
 [*Area* from crisis to crisis].

(frame *Self_motion*). Such phenomena could only be dealt with if subgraph span matching were relaxed to allow multiple graph nodes as role fillers, which we expect would considerably lower its sensitivity to parser errors.

As an example of a role coinciding with the verbal FEE, consider

[*Sound_source* A bat] [*FEE, Sound* squeaked]
 [*Place* over our heads].

(frame *Make_noise*). To cover this case, we would have to match the role span of the role *Sound* to the node span instead of the subgraph span of *squeaked*. Careful adaptation to some of these phenomena could probably increase the part of FrameNet covered with the Span Comparison method by a few percent. For the SALSA corpus, the situation is similar, with non-constituent (or multi-constituent) roles responsible for more than half of the failures on correctly parsed sentences. Here careful treatment e.g. of support verb constructions could also yield some improvement. The failures of the Node Correspondence method can mostly be attributed to either occasional conversion errors or (for the hybrid generation approach) mismatches in converted and reparsed structures, resulting in erroneous correspondences.

5. Conclusion

We presented two methods for accurately enriching Frame Semantic resources with dependency graphs, one using analyses provided by a dependency parser and one based on converted graphs with correspondence links to annotated phrase structure trees. The methods were applied to the FrameNet and the SALSA corpus. Several combinations of those methods with different ways of generating dependency graphs were explored.

For parser-based approaches up to 65.1% (FrameNet) or 64.5% (SALSA) of the corpora were covered. The conversion-based approach covered almost all of the SALSA corpus, while the hybrid approach, adding deep

parser information to the resulting dependency graphs, attained a coverage of 85.4%. Accuracy of the enriched resources was estimated to range from around 80% for approaches involving parsing to more than 90% in the case of treebank conversion. Parser errors were found to be a major reason for failures, showing that our method acts as a good filter on them. Analysis of the remaining failures showed that small improvements could be attained by careful adaptation of the method to particular annotation schemes, an approach which might be worthwhile for practical applications, even if it reduces general applicability of the method.

6. Acknowledgements

The work presented in this paper was supported by the DFG IRTG “Language Technology and Cognitive Systems”. Many thanks to Martin Forst for help with treebank conversion and matching of LFG parses, and to Caroline Sporleder for helpful comments.

7. References

- Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the COLING-ACL*, Montreal, Canada.
- E. Briscoe, J. Carroll, and R. Watson. 2006. The Second Release of the RASP System. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, Sydney, Australia.
- A. Burchardt, K. Erk, A. Frank, A. Kowalski, S. Pado, and M. Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of LREC 2006*, Genoa, Italy.
- Stefanie Dipper. 2003. Implementing and Documenting Large-Scale Grammars — German LFG. In *Arbeitspapiere des Instituts für Maschinelle Sprachverarbeitung (AIMS), Vol. 9(1)*. Doctoral Dissertation, IMS, University of Stuttgart.
- Charles J. Fillmore. 1982. Frame semantics. In *Linguistics in the morning calm*, pages 111–137. Hanshin, Seoul, Korea.
- Martin Forst. 2003. Treebank Conversion - Establishing a test suite for a broad-coverage LFG from the the TIGER treebank. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora (LINC '03)*, Budapest, Hungary.
- Martin Forst. 2007. Filling Statistics with Linguistics - Property Design for the Disambiguation of German LFG Parses. In *Proceedings of the ACL Workshop on Deep Linguistic Processing*, Prague, Czech Republic.
- Anette Frank and Jiří Semecký. 2004. Corpus-based Induction of an LFG Syntax-Semantics Interface for Frame Semantic Processing. In Silvia Hansen-Schirra, Stefan Oepen, and Hans Uszkoreit, editors, *Proceedings of the 5th International Conference on Linguistically Interpreted Corpora, LINC 2004*, Geneva, Switzerland.
- Kadri Hacioglu. 2004. Semantic Role Labeling Using Dependency Trees. In *Proceedings of COLING 2004*, Geneva, Switzerland.
- Alessandro Moschitti, Paul Morarescu, and Sanda Harabagiu. 2003. Open-Domain Information Extraction

- via Automatic Semantic Labeling. In *Proceedings of FLAIRS 2003*, St. Augustine, FL.
- Stefan Riezler, Tracy H. King, Richard S. Crouch, and Annie Zaenen. 2003. Statistical Sentence Condensation using Ambiguity Packing and Stochastic Disambiguation Methods for Lexical-Functional Grammar. In *Proceedings of the Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'03)*.
- Christian Rohrer and Martin Forst. 2006. Improving coverage and parsing quality of a large-scale LFG for German. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy.
- Dan Shen and Mirella Lapata. 2007. Using Semantic Roles to Improve Question Answering. In *Proceedings of EMNLP 2007*, Prague, Czech Republic.