On deriving semantic representations from dependencies: A practical approach for evaluating meaning in learner corpora

Michael Hahn **Detmar Meurers**
Seminar für Sprachwissenschaft
Universität Tübingen
{mhahn,dm}@sfs.uni-tuebingen.de

Abstract

One of the reasons for the popularity of dependency approaches in recent computational linguistics is their ability to efficiently derive the core functor-argument structure of a sentence as an interface to semantic interpretation. Exploring this feature of dependency structures further, in this paper we show how basic dependency representations can be mapped to semantic representation as used in Lexical Resource Semantics (Richter and Sailer 2003), an underspecified semantic formalism originally developed as a semantic formalism for the HPSG framework (Pollard and Sag 1994) and its elaborate syntactic representations.

We describe a two stage process, which in the first stage establishes a syntax-semantics interface representation abstracting away from some differences in surface dependencies. It ensures the local reconstruction of arguments for middle and long-distance dependencies, before building the actual LRS semantics in the second stage. We evaluate the approach on the CREG-109 corpus, a small dependency-annotated corpus with answers to reading comprehension questions written by American learners of German.

1 Introduction

Computational linguistics in recent years has seen a rise in interest in tasks that involve the evaluation and comparison of meaning. A good example is the research strand which has developed around the Recognizing Textual Entailment Challenge (Dagan et al. 2009). Dependency representations have received a lot of attention in this context given that they provide access to functor-argument structures without requiring a computationally costly commitment to a more complex syntactic constituent structure.

In our work, the task is the evaluation of answers to reading comprehension questions. We want to determine, whether the answer given by a student expresses the same meaning as that expressed in a target answer given by the teacher. Such a comparison can be carried out at different levels of abstraction, starting with direct comparisons of the surface forms of words, as, for example, used in the BLEU and ROUGE metrics for evaluating machine translation and summarization approaches (Lin and Och 2004; Lin 2004). At the other extreme are comparisons on the basis of deep semantic analysis and logical inference, which, however, in practice do not necessarily outperform the shallow methods (Bos and Markert 2006). For exploring the space in-between these two extremes, looking for representations on the basis of which meaning can be compared and how they can robustly be obtained, in this paper we discuss the derivation of underspecified semantic representations on the basis of a dependency analysis.

We make this process concrete on the basis of *Lexical Resource Semantics* (LRS, Richter and Sailer 2003), an underspecified semantic formalism which provides explicit semantic representation while at the same time exposing all minimal building blocks of the semantics. This is relevant for our overall goal of having access to a rich space of representations for comparing meanings even in situations where no complete semantics can be obtained. Using LRS essentially allows us to (re)use semantic analyses developed in model-theoretic frameworks. The goal of this research is to further the understanding of the mapping and the information needed to derive semantic representations from bare dependency representations – with a specific focus on LRS representations as a kind of normal form facilitating meaning comparison, an abstraction away from the significant well-formed and ill-formed variation exhibited by learner language.

2 Creating lexically enriched syntax-semantics interface representations

To derive semantic LRS representations from dependency structures, we use a two-step approach. In the first step, the syntactic structure is transformed into a syntax-semantics interface representation, from which the semantic contribution of each word can be computed in the second step, independent of the syntactic structure.

Adopting a two-step approach instead of a single-step mapping makes the system more robust and flexible. The feature-based interface representation abstracts away from variation in form and grammaticality. The semantic representation as such then can be built on this strongly constrained interface representation and the system constructing the semantics does not need to take into account the large space of possible underlying dependency configurations. The explicit interface representation thus makes the semantic construction process more robust against unexpected parses. It also makes the procedure building the semantic representation more transparent. As we will show, only a small number of rather simple rules is needed. And implementing new semantic analyses is rather straightforward if the relevant information is encoded in the interface representation. Last but not least, the interface representation also allows us to build on previous work on semantics within the HPSG architecture since the system can model the HPSG syntax-semantics interface on the lexical level directly.

2.1 The nature of the representations

We focus on the core local, middle-distance and long-distance relations in a sentence. The goal is to achieve good coverage of language phenomena in general as well as to deal with well-known argument-structure challenges.

The representation used to capture the properties needed to identify these relations are represented by a set of features which are defined for every word. They mainly provide information about valency, modification and a more fine-grained labelling of syntactic categories. The following features are used:

- PRED: the core semantic relation expressed by the word, generally represented by its lemma
- CAT: the syntactic category
- ARGS: the set of arguments with their role labels. Only semantically non-empty arguments are represented. The elements of ARGS are feature structures of the form

$$\begin{bmatrix} \text{ROLE} & \text{(role label)} \\ \text{ARG} & \text{(the argument itself)} \end{bmatrix}.$$

- MOD: a modified word, if there is one
- MODTYPE: type of modification (possessor, time, etc.)
- CONJUNCTS: the conjuncts, if this is a first conjunct
- INTERROGATIVE: *true* if this is an interrogative
- IS-PREDICATIVE: *true* if this is a verb or a predicative argument
- TENSE: the tense of a predicative head

An example interface representation for *schreibt* 'writes' as in (1) is shown in (2):

(1) *Peter schreibt einen Brief.*

Peter	writes	a	letter

'Peter writes a letter.'

(2)
$$\begin{bmatrix} \text{PRED} & \textit{`schreiben'} \\ \text{ARGS} & \left\langle \begin{bmatrix} \text{ROLE} & \textit{subj} \\ \text{ARG} & \begin{bmatrix} \text{PRED} & \textit{`Peter'} \\ \text{CAT} & \textit{noun} \\ \text{ISPRED} & \textit{false} \end{bmatrix} \end{bmatrix}, \begin{bmatrix} \text{ROLE} & \textit{obja} \\ \text{ARG} & \begin{bmatrix} \text{PRED} & \textit{`Brief'} \\ \text{CAT} & \textit{noun} \\ \text{ISPRED} & \textit{false} \end{bmatrix} \end{bmatrix} \right\rangle \\ \text{ISPRED} & \textit{true} \\ \text{TENSE} & \textit{present} \\ \text{CAT} & \textit{verb-finite} \end{bmatrix}$$

Extracting this information involves the recursive processing of dependencies and also identifying where dislocated elements are to be interpreted. Some of the features are straightforward to specify locally. CAT, INTERROGATIVE, for example, can simply be assigned based on the dependency labels and part-of-speech tag in the input. However, a well-investigated set of linguistic phenomena, such as strong and weak unbounded dependencies and non-finite constructions, involves dependents which are not realized locally. Our system starts out by building interface representations for local dependency relations only. The structure is then transformed by a procedure which tries to reconstruct a correct representation by moving, copying and adding arguments to the representations of different heads.

2.2 Argument structure challenges

German, as the language we are focusing on here, includes several phenomena which cause arguments to be realized separate from the head they belong to semantically. These include fronting, extraposition, raising, control, passive, and the so-called coherent constructions of the verbal complex (Bech 1955). Since relative pronouns and relative clauses are marked in the dependency parse, identifying relative pronouns with their antecedent can be achieved by recursively searching the dependency graph for a dependency whose dependent is labelled as relative clause and which dominates a given relative pronoun. Other extraction phenomena and the interaction of raising, control, passive and the German verbal complex are more complex to handle since they can interact to form sequences of several verbs, with sequences of three or four verbs being relatively common in written text. While an in-depth discussion of these phenomena clearly is beyond the scope of this paper (cf. Meurers 2000 for an overview), let us illustrate the issue with two examples. Sentence (3) shows a basic example including a future perfect construction and a modal verb.

(3)							
	dass	*ihn*	*Peter*	[*wird*	[*haben*	[*treffen*	*können*]]]
	that	him	Peter	will	have	meet	be able to

'that Peter will have been able to meet him.'

Here, *Peter* is interpreted as the subject of *treffen* 'meet', yet it must also be identified as the subject of the equi predicate *können* 'be able to' which is raised further to become the syntactic subject of the perfect tense auxiliary *haben* and to finally be realized as the subject of the future auxiliary *wird*, which shows subject-verb agreement with it. Similarly, *ihn* 'him' is interpreted as the object of *treffen* 'meet', but given that the other predicates all construct in a coherent verbal cluster, it is ultimately realized as a syntactic argument of the matrix verb *wird* 'will' together with the raised subject *Peter*.

That there is indeed a complex interaction of the different types of lexically triggered argument sharing phenomena going on that needs to be captured can readily be illustrated with the so-called long-distance passivization (Höhle 1978, pp. 175ff) shown in (4).

(4)						
	wenn	*der*	*Wagen*	[[[*zu reparieren*]	*versucht*]	*wird*]
	when	the$_N$	car	to repair	tried	is

'when it is attempted to repair the car'

Here, the passive auxiliary *wird* 'is' selects the verbal complement *versuchen* 'try to', which however is not a verb selecting an NP object that could be promoted to become the subject. Instead, *versuchen* selects the verbal argument *reparieren* 'to repair'. Since this is a coherent verbal complex, the argument of *reparieren* also becomes a syntactic dependent of *versuchen* and as such can then be lifted up by the passive auxiliary *wird* to become the subject of the sentence, with nominative case and showing subject-verb agreement with the finite verb.

Building an adequate interface representation clearly requires lexical information about the verbs selecting nonfinite complements. This includes knowledge about whether a verb is a raising or equi predicate and what its orientation is, i.e., which argument slot the raised or controlled subject fills. Furthermore, we need to know which arguments of its own such a verb selects (as, e.g., the dative NP object required by the subject control equi verb *versprechen*).

A basic reconstruction algorithm The procedure for reconstructing functional structures is based on the general observation that all argument sharing constructions involve a predicate which specifies something about the dependents of its verbal complement. A reconstruction algorithm thus only has to increase the depth of embedding of arguments, but never has to decrease them. Therefore, reconstruction starts from the least embedded verb. Some arguments are moved or copied to the ARGS list of the nonfinite argument, and the same procedure is applied recursively to the embedded predicate, until a predicate without a nonfinite or predicative complement is reached. We furthermore assume that the decision to move an argument can be made locally and depends only on the two verbs under consideration.

In each recursive step, the embedded predicate is identified by its function label PRED, OBJ, or AUX. If the dependency parse is correct and the sentence grammatical, at most one such argument will be present. If no or more than one are found, the algorithm stops. Else, the following operations are carried out:

1. If the matrix verb is *not a passive marker*, the argument with the role label matching the verb's orientation is selected and copied to the ARGS list of the embedded verb, where it has role label *subject*. If the matrix verb is a raising predicate, the copied dependent is deleted from its ARGS list.

2. If the matrix verb is a *tense-marking auxiliary*, the TENSE value of the embedded verb is updated.

3. All arguments which do not match a slot in the verb's argument frame are moved to the ARGS list of the embedded verb. If the surplus arguments cannot be unambiguously determined, no argument is selected.

4. If the matrix verb is the passive auxiliary *werden* or the dative passive marker *bekommen* and the embedded verb is a passive participle, the subject becomes an object of the embedded verb. If a *von* ('by') PP is available, which might encode the agent, its relation is changed to *von_or_subj*. Otherwise, an unspecified subject is added.

As an example, we apply this procedure to the long-distance passivization example we saw in (4) in the way illustrated in (5). The example shows the ARGS lists before reconstruction (a) and after the two recursive steps (b-c).

(5) a.
$$\boxed{1}\begin{bmatrix}\text{PRED} & \textit{werden}\\ \text{ARGS} & \langle \textit{AUX}\ \boxed{2},\ \textit{SUBJ Wagen}\rangle\end{bmatrix}$$
$$\boxed{2}\begin{bmatrix}\text{PRED} & \textit{versuchen}\\ \text{ARGS} & \langle \textit{OBJI}\ \boxed{3}\rangle\end{bmatrix}$$
$$\boxed{3}\begin{bmatrix}\text{PRED} & \textit{reparieren}\\ \text{ARGS} & \langle\rangle\end{bmatrix}$$

b.
$$\boxed{1}\begin{bmatrix}\text{PRED} & \textit{werden}\\ \text{ARGS} & \langle \textit{AUX}\ \boxed{2}\rangle\end{bmatrix}$$
$$\boxed{2}\begin{bmatrix}\text{PRED} & \textit{versuchen}\\ \text{ARGS} & \langle \textit{OBJI}\ \boxed{3},\ \textit{OBJ Wagen},\ \textit{SUBJ PRO}\rangle\end{bmatrix}$$
$$\boxed{3}\begin{bmatrix}\text{PRED} & \textit{reparieren}\\ \text{ARGS} & \langle\rangle\end{bmatrix}$$

c.
$$\boxed{1}\begin{bmatrix}\text{PRED} & \textit{werden}\\ \text{ARGS} & \langle \textit{AUX}\ \boxed{2}\rangle\end{bmatrix}$$
$$\boxed{2}\begin{bmatrix}\text{PRED} & \textit{versuchen}\\ \text{ARGS} & \langle \textit{OBJI}\ \boxed{3},\ \textit{SUBJ}\ \boxed{4}\ \textit{PRO}\rangle\end{bmatrix}$$
$$\boxed{3}\begin{bmatrix}\text{PRED} & \textit{reparieren}\\ \text{ARGS} & \langle \textit{OBJ Wagen},\ \textit{SUBJ}\ \boxed{4}\ \textit{PRO}\rangle\end{bmatrix}$$

In the first step, resulting in (5b), the passive marker *wird* is dealt with, for which *Wagen* is removed and turned into the subject of the passivized verb *versucht*. It has no overt agent, therefore a *pro* subject is added. In the second step, resulting in (5c), the subject control equi verb *versuchen* is considered and its subject is copied to the ARGS list of *reparieren*. The accusative object *Wagen* does not match an argument slot in the lexical entry of *versuchen* and is moved to the embedded verb. The verb *reparieren* does not embed a predicate so that the algorithm terminates.

To some extent, this procedure can also deal with fronting in V2 clauses and with relative clauses. However, the lexical information only allows handling dislocated arguments – the correct attachment of adjuncts cannot be determined.

2.3 Relation to other formalisms

Our interface representations are related to LFG f-structures (Kaplan and Bresnan 1995). Most of our features directly translate into common f-structures features. However, our interface representations differ from standard assumptions about f-structures in that they are closer to the underlying argument structure, i.e., the LFG a-structure. In the interface representations of passive verbs, the agent has the role SUBJ and the patient roles like OBJA. Non-thematic subjects and complements are not represented. This treatment allows a straightforward analysis of some aspects of German syntax such as long-distance passivization, as we will show below. Furthermore, semantic composition is simpler than in LFG, since the arguments represented in the interface representation of some word are always exactly those having a semantic role.

Our two-step approach is also similar to some aspects of the architecture of Meaning Text Theory (Mel'cuk 1988). Our interface representations can be compared to Deep Syntactic Structure, as it also acts as the interface between the surface syntactic dependency structure and a deep semantic representation. While we chose a feature-structure based representation for interface representations, our features ARGS, MOD, MODTYPE and CONJUNCTS can be seen as direct encodings of labelled dependency arcs. However, our interface representations differ from Deep Syntactic Structure in Meaning Text Theory in that they are invariant under phenomena such as passivization, which are already encoded in Deep Syntactic Structure.

The representations are also reminiscent of the linguistic encodings used in HPSG (Pollard and Sag 1994), in particular the treatment of adjuncts as selecting their heads by the MOD feature, which is useful for lexicalized semantic composition. The ARGS list is related to the ARG-ST list often assumed in HPSG, which can be seen as representing the underlying argument structure (Manning and Sag 1998). Furthermore, it appears that all the information contained in our representations is inherent in HPSG analyses and could easily be automatically extracted.

3 The semantic formalism: LRS

Lexical Resource Semantics (LRS, Richter and Sailer 2003) is an underspecified semantic formalism which embeds model-theoretic semantic languages like Ty2 into typed feature structures as used in HPSG. It is formalized in the *Relational Speciate Reentrancy Language* (RSRL, Richter 2000). While classical formal semantics uses fully explicit logical formulae, the idea of underspecified formalisms such as LRS is to derive semantic representations which are not completely specified and subsume a set of possible resolved expressions, thus abstracting away from scope ambiguities.

While other underspecified formalisms used in HPSG such as MRS (Copestake et al. 2005) encode only an underspecified representation, whose relation to resolved representations is external to the representation language, an LRS representation includes both a resolved representation and a representation of its subexpressions, on which scope constraints can be expressed by the relation ◁ *'is a subexpression of'*.

An *lrs* object has three features: INCONT (INTERNAL CONTENT) encodes the core semantic contribution of the head, EXCONT (EXTERNAL CONTENT) the semantic representation of the head's maximal projection, and PARTS is a list containing the subterms contributed by the words belonging to the constituent. An example is given in (6), a semantic representation for *schreibt* in (2).

(6) a. $$\begin{bmatrix} \text{INCONT} & \boxed{1}\ \mathit{schreiben}'(e) \\ \text{EXCONT} & \boxed{2} \\ \text{PARTS} & \left\langle \begin{array}{l} \exists e[\boxed{3} \wedge \boxed{4}], \\ \boxed{7}\mathit{present}(\hat{}\,\boxed{5}), \\ \boxed{6}(\boxed{1}\mathit{schreiben}'(e) \wedge \\ \mathit{subj}(e, \mathit{peter}) \wedge \mathit{obj}(e, y)), \\ \ldots \end{array} \right\rangle \end{bmatrix}$$

b. $\boxed{6} \lhd \boxed{2} \wedge \boxed{6} \lhd \boxed{3} \wedge \boxed{6} \lhd \boxed{5}$

The INCONT value $\mathit{schreiben}'(e)$ is the core semantic contribution. The value of EXCONT is not specified, because it also contains the semantics of arguments and modifiers of the verb. The PARTS list contains three 'maximal' terms: $\exists e[\boxed{3} \wedge \boxed{4}]$ is the quantifier for the event variable, $\mathit{present}(\hat{}\,\boxed{5})$ is the semantic representation of tense marking and $\boxed{6}(\boxed{1}\mathit{schreiben}'(e) \wedge \mathit{subj}(e, x) \wedge \mathit{obj}(e, y))$ represents the verb with its argument structure. Furthermore, PARTS contains every one of their subexpressions with the exception of those which are contributed by another word, but they are omitted in the figure for reasons of readability. The three subexpression constraints in (6b) ensure that the core semantic contribution and the specification of the arguments is part of the representation of the maximal projection, that the event variable is bound by a quantifier, and that the tense predicate outscopes the core semantic contribution.

A possible resolved value for EXCONT of *schreibt* in example (1) is shown in (7).

(7) $\boxed{2}(\exists y[brief(y) \wedge \boxed{7}present(\,\hat{}\,\exists e[\boxed{5}\boxed{6}(\boxed{3}\boxed{1}$ $schreiben'(e) \wedge subj(e, peter) \wedge obj(e, y))]])$

All elements of PARTS are subterms of the complete representation and the subexpression constraints are satisfied.

Unlike some other implementations of deep semantic frameworks, LRS does not employ the lambda calculus as its combinatorial mechanism. Instead, a grammar with an LRS semantics contains three sets of constraints linking syntax and semantics. The INCONT PRINCIPLE ensures that the core semantic contribution (INCONT) is part of the representation of the maximal projection and lexically contributed by the word. The EXCONT PRINCIPLE essentially states that all semantic expressions have to be introduced lexically via the PARTS list. The SEMANTICS PRINCIPLE is grammar-dependent and we show only one exemplary clause:

- INCONT PRINCIPLE:

  INCONT is a subterm of EXCONT and a member of PARTS.

- EXCONT PRINCIPLE:

  In a maximal projection, EXCONT is a member of PARTS.

  In an utterance, α is a member of PARTS iff it is a subexpression of EXCONT.

- SEMANTICS PRINCIPLE:

  – If the nonhead is a quantifier, then the INCONT value of the head is a component of the restrictor.

  – ...

Adapting LRS for Interface Representations LRS was originally developed for constituency-based grammars such as HPSG, and the combinatorial constraints make reference to phrasal notions such as *maximal projection*. Nevertheless, the formalism can easily be used for our syntax-semantics interface representations or standard dependency representations. Unlike other underspecified formalisms used in HPSG such as MRS (Copestake et al. 2005), LRS is strictly lexicalized in the sense that all subexpressions of the complete semantic representation have to be introduced at the word level, and INCONT and EXCONT are the same in all projections of a head. Therefore, combinatorial constraints in the SEMANTICS PRINCIPLE which make reference to *non-maximal* projections can straightforwardly be reformulated in terms of dependencies or the features ARGS and MOD. Representations on the level of nonmaximal projections are not necessary for the combinatorial mechanisms of LRS to work.

The EXCONT PRINCIPLE refers to the elements PARTS list of maximal projections, but this can be replaced by referring to the union of the semantic contributions of the direct and indirect dependents. Technically, this can be implemented in the feature-structure-based LRS formalism by a second list DOMAIN-PARTS which is defined recursively as the concatenation of PARTS and the DOMAIN-PARTS lists of all dependents of the word. Thus, all combinatorial constraints of LRS can be translated into lexicalized, dependency-like formalisms such as our interface representations.

In the next section, we will show how INCONT and PARTS values on the lexical level can be obtained from interface representations.

4 Building LRS representations

For building the LRS representation, only the interface representation built in the first step is required. Building the semantic representation is completely local and rather straightforward since all the required information is included in the interface representation.

In the beginning, INCONT and EXCONT are initialized to unspecified objects; PARTS to the empty list. This structure is successively built up to a full semantic representation by applying rewrite rules, which can be applied in any order. Each rule consists of a condition which is a partial description of the syntactic representation and a consequence, which is a set of operations for adding information to the semantic representation. These operations include: *identifying two objects*, *adding ob-*

jects to PARTS, and *adding subexpression constraints.* In the following, we discuss some exemplary rules to illustrate the nature of the procedure. We will use the name of a feature to refer to its value, e.g., TENSE($\hat{}\,\alpha$) denotes the application of a function with the TENSE value as its name on $\hat{}\,\alpha$. The semantic representations given are a selection of items from PARTS and in some cases relevant subexpression constraints. The word to which the rule applies and the terms added by the rule are printed in boldface.

cat = *verb*: Besides a term defining the predicate, such as *schreiben(e)*, where e is the event variable, and a quantifier binding the variable, terms relating the event variable and the semantic arguments are introduced. If the argument is marked as predicative, the term $R(e, \hat{}\,\alpha)$ is added, where R is the role label of the argument. α is constrained to contain the EXCONT-value of the argument. Otherwise, the term is simply $R(e, x)$, where x is the variable associated with the argument. (6) illustrates this rule. As the figure shows, the PARTS list also contains the subexpressions of the terms added.

cat = *aux*: Since the semantically relevant information was already transported to embedded predicates, auxiliaries are not interpreted at all. Their PARTS list is empty and their INCONT and EXCONT values are equated with those of their complement.

cat = *preposition*: The treatment of prepositions is designed to maximize the invariance of the semantic representation with regard to the variation between adjunct and argument PPs, between argument PPs and argument NPs, and between PPs and pro-forms such as *dahin* 'thither' and *woher* 'whither', which also receive CAT *preposition* in the syntactic analysis. Adjunct and argument PPs are assimilated already in the interface representation, where it is assumed that all prepositions select the head by MOD. The INCONT value of a preposition is always PRED(A_1, A_2). If the ARGS list does not contain a complement, A_2 is set to a new variable, i.e., as the referent of a pronoun or as variable bound by an interrogative quantifier, which is built by a different rule operating on all interrogatives. If there is in argument, A_2 is a first- or higher-order expression as explained for arguments of verbs. A_1 is the index of either the MOD value or the subject. Some aspects of the representation are illustrated by these examples:

(8) *Hans war* ***im*** *Haus.*
Hans was in.the house
'Hans was in the house'

$\langle$***in(hans, x)***, $haus(x), past(\hat{}\,\alpha)$, ...$\rangle$
with $in(hans, x) \lhd \alpha$

(9) ***Wohin*** *geht Hans?*
where goes Hans
'Where does Hans go to?'

$\langle gehen(e) \wedge subj(e, hans) \wedge$
wohin(e,x), $interrog\_q\ x\ \alpha$, ...$\rangle$

(10) *Hans geht* ***nach*** *Berlin.*
Hans goes to Berlin
'Hans goes to Berlin.'

$\langle gehen(e) \wedge subj(e, hans) \wedge$
nach(e,berlin), ...$\rangle$

cat = *adverb*, **mod** $\neq$ *none***:** The INCONT value of adverbial modifiers is PRED($\hat{}\,\alpha$) with MOD|INCONT $\lhd\ \alpha$, i.e., they outscope the core semantic contribution of the verb, while the relative scope of modifiers is not specified.

cat = *noun*, **mod** $\neq$ *none***:** For nominal modifiers, the term MODTYPE(MOD|INDEX, INDEX) is added. This for example accounts for:

(11) *das Buch des* ***Kindes***
the book the.GEN child.GEN
'the child's book'

$\langle$***POSS(x, y)***, $buch(x), kind(y)$,
$def\_q\ x\ [\alpha \circ \beta], def\_q\ y\ [\gamma \circ \delta]$, ...$\rangle$

(12) *Hans kochte zwei* ***Stunden***
Hans cooked two hours
'Hans cooked for two hours'

$\langle kochen(e) \wedge subj(e, hans)$, ***TIME(e,x)***,
$stunde(x), 2\ x\ [\alpha \wedge \beta]$, ...$\rangle$

tense $\neq$ *none***:** The term TENSE($\hat{}\,\alpha$) with INCONT$\lhd\alpha$ is added. Note that also predicative NPs and PPs will receive tense marking, as *Peter* in (13):

(13) *Hans war* ***Peter***.
Hans was Peter
'Hans was Peter.'

$\langle$***PAST***$(\hat{}\,\alpha), hans = peter$, ...$\rangle$
with $hans = peter \lhd \alpha$

The total system consists of 22 rules building the semantic representation from interface representations. Besides basic head-argument and head-modifier structures, some of the covered phenomena are the verb complex, fronting in V2 sentences, relative clauses, coordination and interrogatives. Phenomena which we have not implemented yet include extraposition, ellipsis, focus-sensitive modifiers and discontinuous realization of NPs.

5 Experiment

5.1 Setup

To evaluate the quality and robustness of the systems, we ran two experiments on a small German learner corpus. In the first experiment, we ran the system on a manual dependency annotation and evaluated the resulting LRS structures. To evaluate the meaning of an ungrammatical learner sentence, we constructed a grammatical target hypothesis and then compared it with the automatic semantic analysis. Usually, only one possible analysis was deemed correct, with the exception of adverbs or adjectives modifying verbs, where both an intensional representation (e.g., $really(\hat{}\,come(e)))$ and a representation using the verb's event variable $(real(f) \wedge subj(f, e))$ were admitted. In a second experiment, we ran the same procedure on automatic parses obtained from the statistical MaltParser (Nivre and Hall 2005) trained on Tüba-D/Z (Telljohann et al. 2004) to test the robustness against parsing errors.

5.2 The corpus used

Starting point is the CREG-109 corpus created by Ott and Ziai (2010), a sub-corpus of the Corpus of Reading Comprehension Exercises in German (CREG, Meurers et al. 2010). It consists of 109 sentences representing answers to reading comprehension exercises written by US college students at the beginner and intermediate levels of German programs. Of these, 17 sentences were classified as ungrammatical in that they clearly involved errors in word order, agreement, and case government.

The average sentence length is 8.26; the longest sentence has 17 tokens. CREG-109 was manually annotated by Ott and Ziai (2010) according to the dependency annotation scheme of Foth (2006), which distinguishes thirty-four dependency labels.

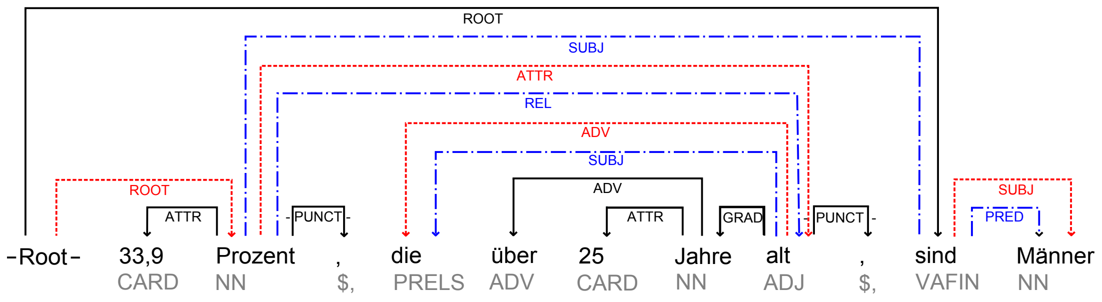
5.3 Results

Using the manual dependency annotation, the semantic representations of 86.2% of the grammatical sentences were fully correct. For 70.5% of the ungrammatical sentences, the analysis was a correct semantic representation of the target hypothesis. Using the automatic parses as input for semantic construction, 65.5% of the grammatical and 47.1% percent of the ungrammatical ones receive a correct representation.

5.4 Error analysis

Apart from ungrammatical input, most errors in the output arise from difficulties with coordination or ellipsis. Problems with coordination are even more severe in the case of automatic parses. Other typical problems caused by noisy parser output are the confusion of subjects and complements, PP-attachment and missing dependencies which isolate some words. The impact of other parser errors on the semantic output is often minor due to the flexibility of the semantic representation language. For example, errors in the attachment of adverbs in the verbal complex are handled to some extent by scope underspecification.

In other cases, even clearly ungrammatical structures receive a correct semantic interpretation, even if an automatic parse which differs from the manual annotation is used. An example for this is given in Figure 1, which Ott and Ziai (2010) give as an example of bad parser performance on ungrammatical input. The dashed blue dependencies are the human annotation, and the red dotted the automatic parse. Inside the relative clause, the copula is missing and the relative clause has no finite verb. The human annotators took the predicative adjective *alt* as the head of the relative clause and *die* as its subject, which yields a correct semantic representation. The parser, on the other hand, interpreted the predicative adjective as adjectival modifier and the relative pronoun as an adverbial modifier. This usage of pronouns is not expected in correct parses and there is no rule dealing with it; therefore, the semantic contribution is empty. Because the noun modified by an adjective is interpreted like an adjective's subject, the ad-


ROOT
SUBJ
ATTR
REL
ADV
SUBJ
ADV
ROOT
ATTR
PUNCT
ATTR
GRAD
PUNCT
SUBJ
PRED
-Root- 33,9 Prozent , die über 25 Jahre alt , sind Männer
CARD NN $, PRELS ADV CARD NN ADJ $, VAFIN NN

Target Hypothesis: *33,9 Prozent, die über 25 Jahre alt [sind], sind Männer.*
33.9 percent who over 25 years old are are men

Figure 1: Parse of an ungrammatical sentence

jective has exactly the same semantic representation. Thus, the correct semantic representation is obtained for the NP *33,9 Prozent, die über 25 Jahre alt.* The example illustrates that abstracting away from the syntactic structure before building the semantic representation can help the system perform well for unexpected syntactic structures which may arise from learner and parser errors.

6 Related work

Spreyer and Frank (2005) use term rewrite rules to build RMRS representations for the TIGER Dependency Bank for German. RMRS is a robust version of Minimal Recursion Semantics, an underspecified semantic formalism used in HPSG. Jakob et al. (2010) present an RMRS system for the Prague Dependency Treebank of Czech. Our work differs in that the input data is learner language and that the semantic representation language is LRS. Furthermore, the dependency parses our system uses contain much less syntactic information than the two dependency banks, in particular no tectogrammatical information.

The first step in our system is related to work on automatically deriving richer feature structure representations such as f-structures from treebank parses (cf. Frank 2000; Cahill et al. 2002). The treebanks used likewise contain more information than the bare dependency parses we use.

7 Conclusion

We presented a system that automatically derives underspecified, model-theoretic semantic representations from dependency parses of German learner sentences. We argued that it is beneficial to first transform dependency structures into syntax-semantics interface representations, which reduce the syntactic structure to semantically important information. In particular, they are invariant under phenomena such as passivization and dislocation. We discussed how such representations can be obtained from dependency parses and presented an algorithm for reconstructing the argument structures of verbs in the German coherent verbal complex, where arguments are commonly realized as dependents of other verbs. We showed that Lexical Resource Semantics, although developed for HPSG, can straightforwardly be adapted to dependency-based syntactic representations, and we presented a sample of a simple rule system building semantic representations in LRS from interface representations. Our evaluation showed that the architecture can often deal robustly with learner and parser errors. In future work, we intend to put these results on a more expressive quantitative basis by evaluating the system on larger native corpora.

References

Gunnar Bech, 1955. *Studien über das deutsche verbum infinitum*. Historisk-filologiske Meddelelser udgivet af Det Kongelige Danske Videnskabernes Selskab. Bind 35, no. 2, 1955; Bind 36, no. 6, 1957; Kopenhagen. Reprinted 1983, Tübingen: Max Niemeyer.

Johan Bos and Katja Markert, 2006. When logical inference helps determining textual entailment (and when it doesn't). In *The Second PASCAL Recognising Textual En-*

tailment Challenge. Proceedings of the Challenges Workshop. pp. 98–103.

Aoife Cahill, Mairead McCarthy, Josef van Genabith and Andy Way, 2002. Parsing with PCFGs and Automatic F-Structure Annotation. In *Proceedings of the LFG-02 Conference*. CSLI Publications.

Ann Copestake, Dan Flickinger, Carl Pollard and Ivan Sag, 2005. Minimal Recursion Semantics: An Introduction. *Research on Language & Computation*, 3:281–332.

Ido Dagan, Bill Dolan, Bernardo Magnini and Dan Roth, 2009. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 15(4):i–xvii.

Kilian Foth, 2006. *Eine umfassende Constraint-Dependenz-Grammatik des Deutschen*. Manual, Universität Hamburg.

Anette Frank, 2000. Automatic F-structure Annotation of Treebank Trees. In *Proceedings of LFG-00*. CSLI Publications.

Tilman N. Höhle, 1978. *Lexikalistische Sxntax. Die Aktiv-Passiv-Relation und andere Infinitkonstruktionen im Deutschen*. Max Niemeyer, Tübingen.

Max Jakob, Marketa Lopatkova and Valia Kordoni, 2010. Mapping between Dependency Structures and Compositional Semantic Representations. In *Proceedings of LREC 2010*.

Ronald M. Kaplan and Joan Bresnan, 1995. Lexical-Functional Grammar: A Formal System for Grammatical Representations. In Mary Dalrymple, John T. Maxwell, Ronald M. Kaplan and Annie Zaenen (eds.), *Formal issues in Lexical-Functional Grammar*, CSLI Publications, Stanford, CA.

Chin-Yew Lin, 2004. Rouge: a package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*.

Chin-Yew Lin and Franz Josef Och, 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the ACL*. Barcelona.

Christopher D. Manning and Ivan A. Sag, 1998. Argument Structure, Valence, and Binding. *Nordic Journal of Linguistics*, 21.

Igor A. Mel'cuk, 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press.

Detmar Meurers, 2000. Lexical Generalizations in the Syntax of German Non-Finite Constructions. Phil. dissertation, Eberhard-Karls-Universität Tübingen. `http://purl.org/dm/papers/diss.html`.

Detmar Meurers, Niels Ott and Ramon Ziai, 2010. Compiling a Task-Based Corpus for the Analysis of Learner Language in Context. In *Proceedings of Linguistic Evidence*. Tübingen, pp. 214–217.

Joakim Nivre and Johan Hall, 2005. Maltparser: A language-independent system for data-driven dependency parsing. In *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories*. pp. 13–95.

Niels Ott and Ramon Ziai, 2010. Evaluating Dependency Parsing Performance on German Learner Language. In *Proceedings of the Ninth International Workshop on Treebanks and Linguistic Theories (TLT9)*.

Carl Pollard and Ivan A. Sag, 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, IL.

Frank Richter, 2000. A Mathematical Formalism for Linguistic Theories with an Application in Head-Driven Phrase Structure Grammar. Phil. dissertation, Eberhard-Karls-Universität Tübingen.

Frank Richter and Manfred Sailer, 2003. Basic Concepts of Lexical Resource Semantics. In Arnold Beckmann and Norbert Preining (eds.), *ESSLLI 2003 – Course Material I*. Kurt Gödel Society, Wien, volume 5 of *Collegium Logicum*, pp. 87–143.

Kathrin Spreyer and Anette Frank, 2005. Projecting RMRS from TIGER Dependencies. In *The Proceedings of the 12th International Conference on HPSG*. CSLI Publications, Stanford, pp. 354–363.

Heike Telljohann, Erhard Hinrichs and Sandra Kübler, 2004. The Tüba-D/Z Treebank: Annotating German with a Context-Free Backbone. In *Proceedings of LREC 2004*. pp. 2229–2235.