Improving DRS Parsing with Separately Predicted Semantic Roles

Tatiana Bladier[1], **Gosse Minnema**[2], **Rik van Noord**[2], **Kilian Evang**[1]
(1) University of Düsseldorf, Germany
(2) University of Groningen, the Netherlands
{tatiana.bladier, evang}@hhu.de
{g.f.minnema, r.i.k.van.noord}@rug.nl

Abstract

This paper addresses Semantic Role Labeling (SRL) within the context of English Discourse Representation Structure (DRS) parsing. In particular, we investigate whether semantic roles predicted by a near-state-of-the-art SRL model can be used to improve the outputs of modern end-to-end neural DRS parsers using a rule-based post-processing algorithm. We compare two methods of generating training data for the SRL model from the Parallel Meaning Bank, one DRS-based and one CCG-based. We also compare two different post-processing algorithms. Our results vary across different DRS parsers, but overall we find a small to moderate improvement of up to 0.5 F1 on the final DRSs. We find a small but consistent advantage of DRS-based over CCG-based training data generation, and of token-based over concept-based post-processing, where applicable.

1 Introduction

With the increasing availability of multi-layered semantically annotated corpora, semantic parsing today is typically approached as an end-to-end task of predicting a meaning representation in one go, including information on word senses, predicate-argument structure, scope, semantic roles, and more. Since each of these layers is complex in its own right, it might be beneficial to rely on multiple specialized components to separately predict individual semantic layers, and to combine their output. In this paper, we focus on separately predicting semantic roles in the context of Discourse Representation Structure (DRS) parsing.

DRSs are meaning representations grounded in Discourse Representation Theory (Kamp and Reyle, 1993). We use the English part of the Parallel Meaning Bank (PMB; Abzianidze et al., 2017), which contains sentences annotated with DRSs. Figure 1 shows an example. Events (e.g., `e1`) are related to their participants (e.g., `x1`, `x2`) via semantic roles (e.g., `Theme`, `Destination`) from the VerbNet/LIRICS inventory (Bonial et al., 2011). Semantic roles are a crucial aspect of meaning since they encode how each entity participates in an event (Fillmore, 1968).

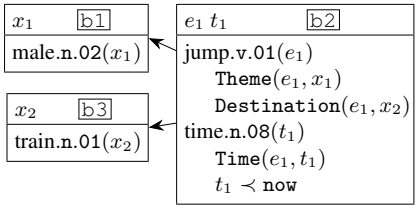

Figure 1: DRS for *He jumped into the train* (source: PMB, document `00/2759`)

Semantic role labelling (SRL) is typically approached as a task of labeling tokens or parse tree edges with predicate/role labels, independently of other aspects of meaning (e.g., Li et al., 2019, 2020b; Shi et al., 2020; Marcheggiani and Titov, 2020; Li et al., 2020a). Conversely, DRS parsers such as Evang (2019); Fancellu et al. (2020); van Noord et al. (2020); Liu et al. (2021) do not have dedicated SRL modules but predict a complete meaning representation of which roles are one part. In this paper, we explore the possibility of combining semantic parsers with a dedicated SRL system. The main research question we seek to answer is: can we in this way obtain DRSs with more accurate semantic roles?

Our approach is summarized in Figure 2: we first convert the PMB training data into a standard SRL annotation format (§2) in order to train a near-state-of-the-art SRL system on it (§3). At test time, we merge the output of DRS parsers with that of the SRL system using a rule-based post-processing algorithm (§4), aiming to produce

Proceedings of the ESSLLI 2021 Workshop on Computing Semantics with Types, Frames and Related Structures, pages 25–34
Utrecht, The Netherlands (online), 26–27 July, 2021.

a more accurate final DRS. We experiment with applying our procedure on top of several recent DRS parsing systems, and find that, albeit with some caveats, our procedure leads to overall better scores (§5). [1]

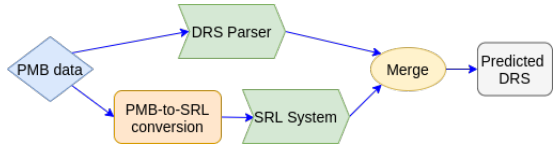

Figure 2: System overview

2 DRS-to-SRL Conversion

Before we can train an SRL system, we first need to convert semantic role annotations in the PMB to a more standard SRL format. Two characteristics of the PMB make this a non-trivial task. First, role annotations in the PMB are *predicate-based*, meaning that roles are carried by predicates instead of by arguments, as in standard SRL systems. Table 1 illustrates this: in standard SRL, the Theme role would be marked on *he*. Instead, in the PMB, the role is annotated on *jumped*, the predicate assigning the role; in a later step, the DRS parser makes sure that the role is associated to the discourse referent introduced by "He". Second, prepositional and adverbial roles (e.g. *into the train*, *slowly*) are treated differently from "core" semantic roles: they are carried by the preposition or adverb itself, instead of by the verbal predicate they are associated to.

Token	*He*	*jumped*	*into*	*the*	*train*
PMB		Theme	Destination		
SRL: head	Theme	PRED			Destination
SRL: span	Theme	PRED	{ ←	Destination	→ }

Table 1: PMB-style versus standard SRL annotations.

We experiment with two approaches for converting PMB role labels to a standard SRL format:

2.1 DRS-based conversion

Here, predicates and fillers for semantic roles are found via DRSs, which in the training data are *anchored*, i.e., most clauses are aligned to exactly one token. We extract predicate-role-filler triples such as ⟨jumped,Theme,he⟩ from the anchored DRSs by looking for role clauses such as `b2 Theme e1 x1` and then finding the clause introducing the filler (`b1 REF x1`, anchored to *He*), and the clause introducing the event (`b2 REF e1`, anchored to *jumped*). The process is illustrated in Figure 3.[2]

Disadvantages of this approach are 1) that it only yields the heads of the fillers, not full spans, and 2) that in some cases, the 'deep' semantic structure of the DRS does not directly match the surface realisations of the semantic roles we want to find. One example of the latter problem is found in sentences such as "She saw herself", where a DRS-based approach would return "She" as the Stimulus role, instead of "herself", which is the surface filler of this role but does not introduce a discourse referent of its own.

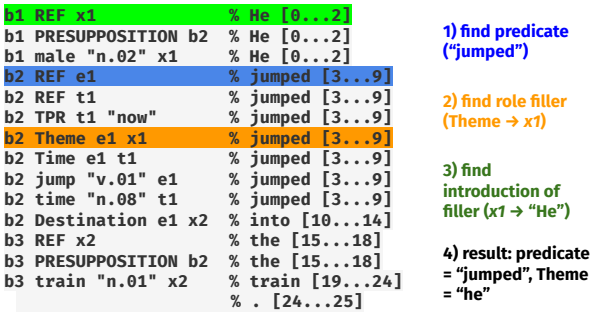

Figure 3: Example of DRS-based conversion.

2.2 CCG-based conversion

The second approach aims at overcoming both limitations of the DRS-based approach by making use of the CCG derivations in the PMB. Here, predicates and fillers for semantic roles are found via the CCG (Categorial Combinatorial Grammar, Steedman 2000) syntax trees and predicate-based role annotations in the PMB.

Main conversion process First, we transform the CCG trees using the `pmb_ccg_to_term` module in the `LangPro` package (Abzianidze, 2017), removing directionality of the combinatory rules and reducing the number of possible combinators, which simplifies tree traversing. In particular, long-distance dependencies (such as *wh*-movement) are handled using the λ-operator, which introduces a relationship between two variables at different points in the tree. An example of this kind of tree is given in Figure 4.

[1] Code and data at https://github.com/TaniaBladier/DRS_Parsing_with_SRL

[2] The DRS in *clause notation* in Figure 3 is equivalent to the one in *box notation* in Figure 1, but additionally shows the alignment with tokens in the sentence.

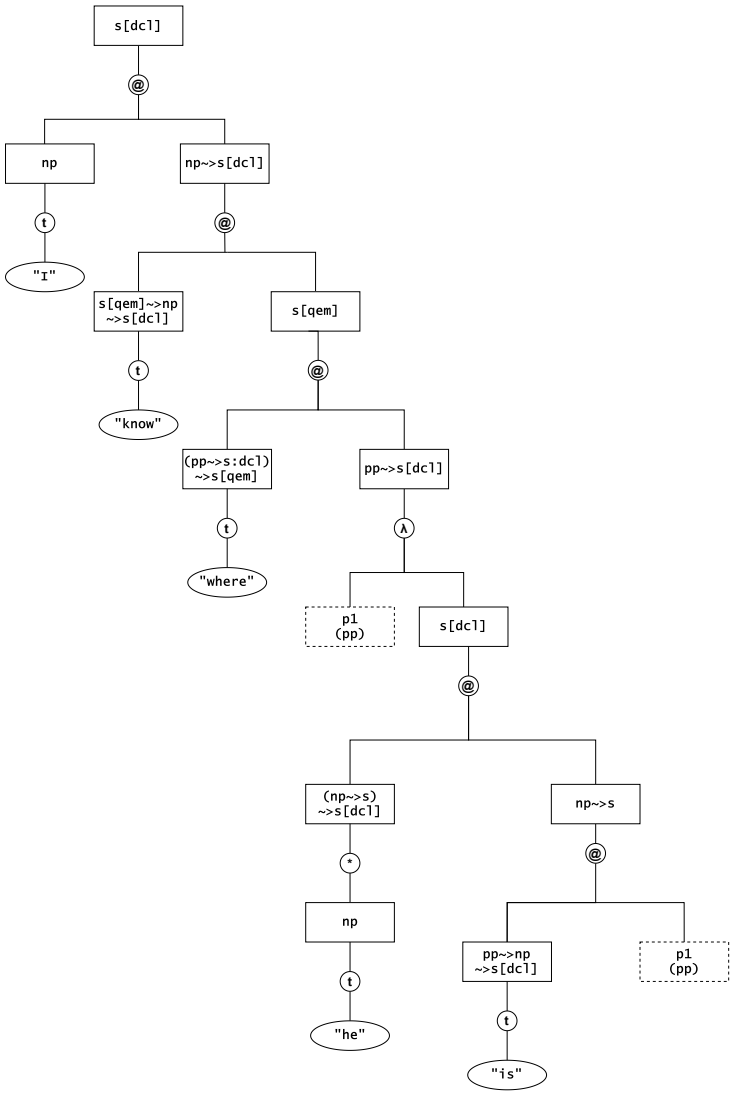

Figure 4: Simplified CCG tree with examples of all combinators (@: simple functional application; λ: variable introduction;, $*$: type-raising). Solid rectangles are types, circles are operators, dotted rectangles are lambda variables, and ovals are lexical nodes. `s[dcl]` means 'declarative sentence'; `s[qem]` means 'embedded question'.

Next, we deploy our role span extraction algorithm, which traverses the simplified tree and tries to match the semantic roles annotated on each predicate to the constituents filling these roles. Figure 5 displays a high-level overview of this process, showing how CCG arguments get mapped to constituents in the tree. This process is explained in more detail in Figure 6.

Given a simplified tree, we extract each predicate's syntactic roles from its CCG type signature and match them with the annotated semantic roles. For example, suppose *jump* has the type signature NP→S[3] and the role annotation `[Theme]`, then it has a single NP syntactic role, corresponding to

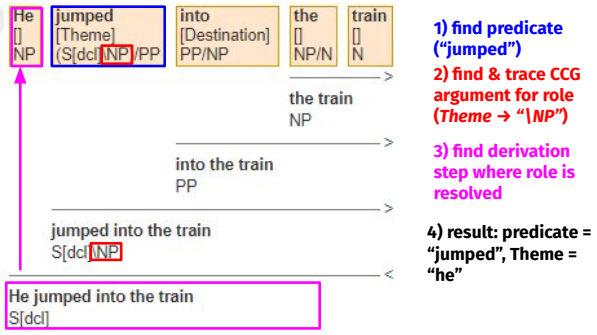

Figure 5: Example of CCG-based conversion.

a Theme semantic role. Then, we move upwards through the syntax tree, checking the type signature at every step; whenever we detect that a role has been filled, we process the constituent that was

[3]The original CCG category would be S\NP, which we simplify into the direction-agnostic NP→S.

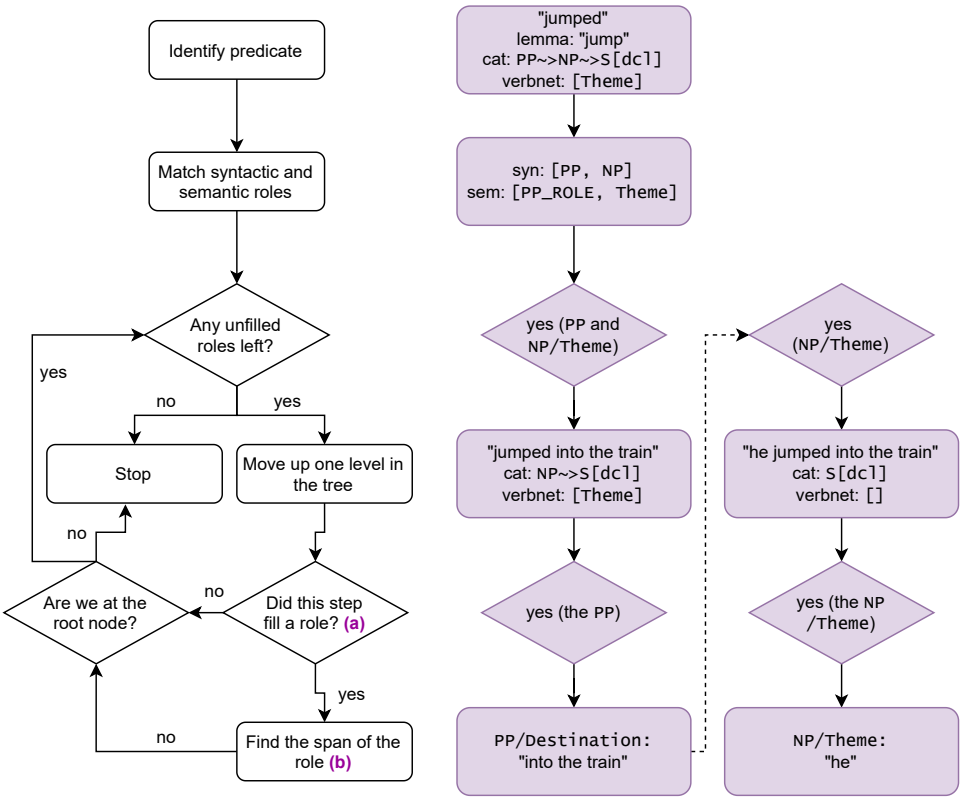

Figure 6: Flow chart of the main CCG-based conversion process. Algorithmic steps in white, example in purple.

merged at that point of the tree as the filler of the corresponding semantic role. This process is repeated until we have found a filler for every role, or until we reach the top of the tree.[4]

Detecting merged constituents A crucial step of our process (step *(a)* in Figure 6) is detecting, given a particular node in the tree, whether a role has been resolved at that node. In many cases, this is straightforward; for example, in the sentence in Figures 5 and 6, we can see that *he* fills the NP/Theme role of *jump* at the point where *he* is combined with *jumped into the train* through simple functional application, changing the type signature from NP→S to S. In other cases, more complicated rules are needed, for example when dealing with to-clauses (*She wants me to leave*), where, on combining *wants me* with *to leave*, the type signature of *to leave* changes from NP→S[to] to NP→S[dcl]. In such cases, at first glance, it appears as if not much has changed except a change of clause type (from a *to*-clause to a declarative sentence), whereas in fact, *me* has filled the subject NP of *leave*, and a new NP argument (the subject NP of *wants*) has been added. We have developed a set of heuristics that cover all such difficult cases occurring in the gold annotations in the PMB. While we believe that this amounts to a wide general coverage, it is likely that there exist other constructions that our algorithm does not (yet) cover.

Once it has been defined that a role is resolved at a given node in the tree, the next crucial step (step *b* in Figure 6) is to find the correct role span within the constituent that was combined. In many cases (like *he* in *he jumped*), the entire constituent is the role filler, but in other cases (like *wants me* in *She wants me to leave*), only a part of the constituent (*me*) is the role filler that we are looking for. To find this constituent, we designed a separate algorithm that moves down the tree starting from the merged constituent, until an argument with the correct type is found.

PP and adverbial roles Semantic roles carried by PP constituents (e.g. *into the train*) or by adverbial phrases (e.g. *quickly*) pose an additional chal-

[4] In some cases, e.g. *wh*-questions, it is possible that some roles remain unfilled.

lenge, since, in the PMB annotation framework, these roles are annotated on the syntactic head of the PP or adverbial phrase (e.g. *into* in *into the train*) rather than on the verb that they combine with. In cases where the PP is a syntactic argument of the verb (as in *jump into the train*), we solve this by first adding a placeholder role (see the `PP_role` at the top of Figure 6) corresponding to the verb's PP argument, and then replacing this by the semantic role carried by the PP at the point where it is combined with the predicate. In cases where a PP or adverb is an adjunct (e.g. with type signature `S→S` or `(NP→S)→(NP→S)`), we add the semantic roles introduced by the adjunct to the predicates in the constituent that is modified (e.g., *quickly* modifies *he ran* in *he ran quickly*. To ensure that adjuncts get the right scope, we added a constraint to our algorithm that forbids adding adjunct roles to predicates if doing so would cross a clause boundary; e.g., *loudly* in *he loudly said he was going to leave* can modify *said* but not *leave*.

Span-to-head conversion As a final step, to make the outputs of the CCG-based algorithm comparable to those of the DRS-based algorithm, we add a final step that converts the extracted role spans to their semantic heads. This algorithm consists of a set of (recursive) rules defining what the head of each type of phrase is. For example, $H(\text{the old woman}) = H(\text{old woman}) = H(\text{woman}) = \text{woman}$, where H is a function applying the appropriate rule for a given phrase type and returning the 'head part' of the phrase. There are many possible phrase types, but in general, the head of an NP is a noun, the head of a VP is a verb, the head of a PP is an NP, and the head of a sentence is the VP.

2.3 Comparing the approaches

Comparing the outputs of both conversion approaches, we find that 68% of documents match exactly, and 82% differ by at most one role. This shows that both approaches show significant differences worth further investigating. The differences mainly concern structural mismatches between syntax and semantics. For example, in sentences with co-referential NPs, CCG-based conversion gives more intuitive results than DRS-based conversion: in *she handed him*$_1$ *the money that she owed him*$_2$, DRS-based conversion treats the two *him*s as the same entity and assigns the Beneficiary role of *owe* to *him*$_1$, whereas CCG-based conversion correctly assigns it to *him*$_2$. Similarly, with reflexives, in *she saw herself*, DRS-based conversion is unable to assign any role to *herself*, since this word does not introduce a new discourse referent but refers back to *she*. The syntax-driven CCG-based conversion also allows for a better resolution of *hearer* and *speaker* discourse participants in such sentences as *I don't remember your name*.

On the other hand, CCG-based conversion has difficulties dealing with light verb constructions where the semantics of the main verb and the light verb interact. For instance, in *he had his wallet stolen*, the relationship between *he* and *stolen* is not detected. Finally, more heuristics will need to be added to CCG-based conversion to cover all adjunct semantic roles due to the way that these are annotated in the PMB, e.g. *by*-clauses in passive sentences. Also, the CCG-based conversion needs additional rules to distinguish between the semantic and syntactic head in such constructions as *all of the town* or *a kilo of plums*.

3 SRL Predictions

We predict semantic roles using the graph-based end-to-end coreference resolution system by He et al. (2018). This syntax-agnostic SRL model jointly predicts predicates, role fillers, and role labels. The SRL system builds contextualized representations for spans of arguments and predicate tokens based on BiLSTM outputs. The argument spans and predicates are predicted independently of each other and the aggressive beam pruning is used to discard the least probable combinations of predicate and argument spans. The output of the system is a graph, which lists predicted SRL roles as edges and the associated text spans as nodes. The SRL graph is predicted directly over text spans. Unlike He et al., we do not predict the full spans of semantic roles, but only syntactic heads of the semantic role spans, since the DRSs in the PMB do not contain information about full spans of arguments.[5] We experiment with GloVe (Pennington et al., 2014) and ELMo (Peters et al., 2018) embeddings to train the SRL system.[6]

We use the gold section of the English PMB data (release 3.0.0) to train and test the SRL system, which contains a train, dev, and test split of

[5]The full spans of semantic arguments can be reconstructed from head spans using syntactic information from dependency graphs (Gliosca and Amsili, 2019).

[6]The hyper-parameters are given in the appendix.

6 620, 885, and 898 documents, respectively. The SRL system is trained on the output of both DRS-to-SRL conversion tools separately. We include only verbal predicates and exclude the predicate *be* due to its inconsistent annotation in the PMB.

4 Merging DRS and SRL Predictions

As baseline DRS parsers without external SRL prediction, we use DRS parsers for which the output is publicly available: the transition-based compositional parser of Evang (2019) and three neural sequence-to-sequence models: the character-level model of van Noord et al. (2018b), an extension of this model that uses linguistic features (van Noord et al., 2019) and the best BERT-based model of van Noord et al. (2020). We refer to these models with E19, N18, N19, and N20.

We propose two methods for merging DRS and SRL output: a *token-based* method for parsers that are lexically anchored (each clause maps to one token), such as E19, and a *concept-based* method for parsers for which this is not the case (N18, N19, N20). Both methods only aim to *replace* roles in the DRS; no new full clauses are inserted.

Token-based merging When the SRL system predicts a predicate-role-filler tuple such as ⟨jumped,Theme,he⟩, we look for a corresponding role prediction in the parser output. A corresponding prediction is a role clause such as `b2 Agent e1 x1`, where the event discourse referent (`e1`) and the filler discourse referent (`x1`) are introduced by the corresponding tokens, i.e., *jumped*, and *he*, respectively. We say that a referent is introduced by a token if the token is anchored to a concept clause for that referent, such as `b2 jump "v.01" e1` or `b1 male "n.02" x1`. In this example, the DRS parser predicted a different role (Agent) than the SRL system (Theme), so we replace the former with the latter.

Concept-based merging Concept-based merging works similarly but does not rely on clauses being anchored to tokens. Instead, concept clauses are *matched* to tokens using corpus-level alignment and lemmatization. We say that a concept clause (e.g., `b1 male "n.02" x1`) *matches* a token (e.g., *he*) if it is observed anchored to the same word anywhere in the full PMB training data (bronze, silver, and gold). We also say that a concept clause (e.g., `b2 jump "v.01" e1`) matches a token (e.g., *jumped*) if there is a string match between the concept and the lemma[7] of the token (*jump*).

Restrictions In order to avoid some incorrect role replacements, we impose the following heuristics to restrict replacement: a role r is not replaced with r' if 1) r is one of the special roles `Time` and `Name`, 2) r' was predicted by the SRL system with $< 50\%$ precision, 3) r' already exists in the same box as r. For concept-based merging, the general concepts `person`, `be` and `entity` are never matched with any input tokens.

5 Experiments and Discussion

The main results of our experiments are shown in Table 2. Overall, we see small but consistent improvements for all parsers, except for N20, the most recent system. It seems that once the parser reaches a certain accuracy it is not straightforward to improve the scores by using an imperfect external system. This is also reflected by the number of replaced roles, which goes down as the parsers get better. Comparing the two conversion methods, we find that DRS-based conversion leads to higher scores. The difference with CCG-based conversion is small, though consistent between setups. In a sense, this is unsurprising given that DRS is also our target representation format. Furthermore, we found that using ELMo outperformed GloVe; while this is unsurprising, it supports the intuition that using a higher quality SRL system leads to more improvement. In other words, any development on the SRL parsing side is likely to lead to better performance on DRS parsing as well. Comparing token-based to concept-based merging on the output of the E19 parser (the only one where it is applicable), it makes more replacements and results in slightly higher accuracy, suggesting an advantage in terms of recall over concept-based merging.

Room for improvement As can be seen in Table 2, SRL performance seems to be a bottleneck; hence, using future, higher-quality SRL systems might also lead to better overall performance of our method. In particular, due to the merging step in our pipeline system, missing roles in SRL predictions are less costly than wrong predictions. Hence, we expect that SRL systems that are optimized for precision rather than for F-score will be more suited for use in our task. Furthermore, we

[7] We use spaCy (Honnibal et al., 2020) for this.

Experiments	SRL		E19-tok		E19		N18		N19		N20	
	dev	test	dev	test	dev	test	dev	test	dev	test	dev	test
Baseline	–	–	81.4 (0)	81.4 (0)	81.4 (0)	81.4 (0)	84.3 (0)	84.9 (0)	86.8 (0)	88.7 (0)	88.4 (0)	89.3 (0)
DRS conv.: upper	100	100	+1.5 (154)	+1.3 (144)	+1.3 (124)	1.2 (124)	+0.9 (92)	+1.2 (132)	+0.9 (88)	+1.1 (117)	+0.5 (51)	+0.7 (76)
CCG conv.: upper	100	100	+1.2 (145)	+1.2 (134)	+1.2 (115)	1.1 (118)	+0.9 (89)	+1.2 (129)	+0.8 (80)	+1.1 (114)	+0.5 (50)	+0.8 (78)
DRS conv. + GloVe	79.7	81.6	+0.3 (129)	+0.3 (113)	+0.4 (97)	+0.2 (102)	+0.2 (68)	+0.4 (92)	+0.1 (64)	+0.2 (90)	-0.2 (57)	-0.1 (70)
DRS conv. + ELMo	**85.8**	86.3	**+0.5** (128)	**+0.4** (120)	**+0.5** (104)	**+0.4** (110)	**+0.3** (73)	**+0.5** (107)	**+0.2** (74)	**+0.3** (104)	-0.1 (55)	0.0 (69)
CCG conv. + GloVe	80.7	83.0	+0.3 (129)	+0.3 (117)	+0.3 (107)	+0.2 (108)	+0.1 (96)	+0.4 (102)	0.0 (93)	+0.1 (103)	-0.2 (73)	-0.1 (74)
CCG conv. + ELMo	85.2	**87.0**	+0.4 (118)	**+0.4** (109)	+0.4 (99)	+0.3 (103)	+0.2 (81)	+0.4 (104)	+0.1 (73)	+0.2 (102)	-0.2 (63)	0.0 (66)

Table 2: Experiment results, including F-scores and number of replaced roles (in brackets). The F-scores are calculated using Counter (van Noord et al., 2018a). Scores for N19 and N20 are averaged over 5 runs. E19-tok uses token-based merging, E19 uses concept-based merging like the rest.

expect that further improvements in the conversion algorithms will lead to better overall performance.

Error analysis We identified four sources of errors in the SRL predictions. The data show an imbalanced role distribution towards the roles Theme and Agent, which take up 52% of all annotations out of 32 semantic roles. This leads to overprediction of these roles by the SRL-labeler. Indeed, for N20 we find that these roles have an insertion precision of $< 50\%$, or in other words, they were more often wrongly inserted than that they correctly replaced a non-matching role. Figure 7 shows the confusion matrix for the most frequent semantic roles.

pred./gold	Agent	Co-Theme	Dest.	Exper.	Loc.	Patient	Source	Stim.	Theme
Agent	337	0	0	5	0	1	0	0	5
Co-Theme	0	54	0	0	0	1	0	0	3
Destination	0	0	33	0	2	0	0	0	0
Experiencer	1	0		62	0	3	0	1	1
Location	0	0	0	0	62	0	0	0	0
Patient	2	0	1	1		77	0	1	7
Source	1	0	0	0	0	0	21	1	2
Stimulus	2	0	0	0	0	0	0	56	2
Theme	14	2	1	0	2	7	0	4	356

Figure 7: Confusion matrix for semantic labeling errors, showing the numbers of predicted labels for the most frequent labels.

The role Theme and Agent are also frequently predicted extra in cases where no semantic role should be predicted. For example, the pronoun *her* in the sentence *she ate her dinner* is erroneously assigned the role Agent. Semantic roles of prepositional phrases also lead to prediction errors. For example, the phrase *the field of biology* in the sentence *He is working in the field of biology* is wrongly recognized as Location instead of Theme. Another cause of prediction errors are possessive determiners which are wrongly predicted as role fillers. For example, both *her* and *dinner* are predicted as Patient in the following sentence: *She ate her dinner*. Also, no semantic roles are predicted by the SRL-labeler if the head word has no vector embedding due to a special character, for example like *post∼office*. Due to the merging step in our pipeline, the erroneously missing semantic roles in SRL predictions do not lead to a drop of parsing performance and also do not improve it.

6 Conclusions and Future Work

We have presented experiments on using externally predicted semantic roles to improve the output of four recent DRS parsers. We saw that there is considerable room for improvement and our method fills it – but not fully, especially as parsers get more accurate. We conclude that our approach is useful especially with parsers such as E19 which do not reach state-of-the-art accuracy but may have other advantages such as smaller models or lexical anchoring. An advantage of our approach is that it is very flexible: it can be applied on top of any DRS parsing model without having to alter or retrain the model itself. This means that our method, or an improved version of it, could also be applied to future DRS parsers, possibly with completely different architectures. In future work we intend to experiment with enhancing the SRL system using syntactic input from CCG-based supertags and also try out other SRL systems. We also plan to experiment with prediction of nominal and adjectival predicates along with their semantic roles. We also intend to reconstruct and predict full spans of semantic roles. Moreover, we plan to carry out parsing experiments with further languages in the PMB, including Dutch, German, and Italian, as our method should be universally applicable. Finally, it would be interesting to improve the SRL predictions by enforcing coherence of predicted predicates and corresponding semantic roles.

Acknowledgements

We would like to thank two anonymous reviewers for their valuable comments. The work presented in this paper has been partially funded by the European Research Council, within the ERC grant TreeGraSP[8].

References

Lasha Abzianidze. 2017. LangPro: Natural language theorem prover. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 115–120, Copenhagen, Denmark. Association for Computational Linguistics.

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

Claire Bonial, William Corvey, Martha Palmer, Volha V Petukhova, and Harry Bunt. 2011. A hierarchical unification of lirics and verbnet semantic roles. In *2011 IEEE Fifth International Conference on Semantic Computing*, pages 483–489. IEEE.

Kilian Evang. 2019. Transition-based DRS parsing using stack-LSTMs. In *Proceedings of the IWCS Shared Task on Semantic Parsing*, Gothenburg, Sweden. Association for Computational Linguistics.

Federico Fancellu, Ákos Kádár, Ran Zhang, and Afsaneh Fazly. 2020. Accurate polyglot semantic parsing with DAG grammars. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3567–3580, Online. Association for Computational Linguistics.

C.J. Fillmore. 1968. The case for case. In E. Bach and R. Harms, editors, *Universals in Linguistic Theory*. Holt, Rinehart and Winston, New York.

Quentin Gliosca and Pascal Amsili. 2019. Résolution de coréférence basée sur les têtes. In *Actes de la conférence TALN 2019 (articles courts)*, Toulouse.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369, Melbourne, Australia. Association for Computational Linguistics.

[8] https://treegrasp.phil.hhu.de/

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic*. Studies in Linguistics and Philosophy. Kluwer, Dordrecht, Boston, London.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020a. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online. Association for Computational Linguistics.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6730–6737.

Zuchao Li, Hai Zhao, Rui Wang, and Kevin Parnow. 2020b. High-order semantic role labeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1134–1151, Online. Association for Computational Linguistics.

Jiangming Liu, Shay B. Cohen, Mirella Lapata, and Johan Bos. 2021. Universal Discourse Representation Structure Parsing. *Computational Linguistics*, pages 1–33.

Diego Marcheggiani and Ivan Titov. 2020. Graph convolutions over constituent trees for syntax-aware semantic role labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3915–3928, Online. Association for Computational Linguistics.

Rik van Noord, Lasha Abzianidze, Hessel Haagsma, and Johan Bos. 2018a. Evaluating scoped meaning representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Rik van Noord, Lasha Abzianidze, Antonio Toral, and Johan Bos. 2018b. Exploring neural methods for parsing discourse representation structures. *Transactions of the Association for Computational Linguistics*, 6:619–633.

Rik van Noord, Antonio Toral, and Johan Bos. 2019. Linguistic information in neural semantic parsing with multiple encoders. In *Proceedings of the 13th International Conference on Computational Semantics - Short Papers*, pages 24–31, Gothenburg, Sweden. Association for Computational Linguistics.

Rik van Noord, Antonio Toral, and Johan Bos. 2020. Character-level representations improve DRS-based semantic parsing Even in the age of BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4587–4603, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Tianze Shi, Igor Malioutov, and Ozan Irsoy. 2020. Semantic role labeling as syntactic dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7551–7571, Online. Association for Computational Linguistics.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press.

Appendix

Layer	Hyper-parameters	Value
Characters CNN	numb. of filters	50
Bi-LSTM	state size # layers	200 3
Words embedding	vector dim.	300
Char. embedding	dimension	8
	batch size	40
Dropout	dropout rate	0.5
	Max. gradient norm	5.0
	Optimizer	Adam
	Learning rate	0.001
	Decay rate	0.999
	Decay frequency	100

Hyper-parameters of the SRL system.