The Relative Clauses AMR Parsers Hate Most

Xiulin Yang, Nathan Schneider
Georgetown University
Washington, DC, USA
{xy236, nathan.schneider}@georgetown.edu

Abstract

This paper evaluates how well English Abstract Meaning Representation parsers process an important and frequent kind of Long-Distance Dependency construction, namely, relative clauses (RCs). On two syntactically parsed datasets, we evaluate five AMR parsers at recovering the semantic reentrancies triggered by different syntactic subtypes of relative clauses. Our findings reveal a general difficulty among parsers at predicting such reentrancies, with recall below 64% on the EWT corpus. The sequence-to-sequence models (regardless of whether structural biases were included in training) outperform the compositional model. An analysis by relative clause subtype shows that passive subject RCs are the easiest, and oblique and reduced RCs the most challenging, for AMR parsers.

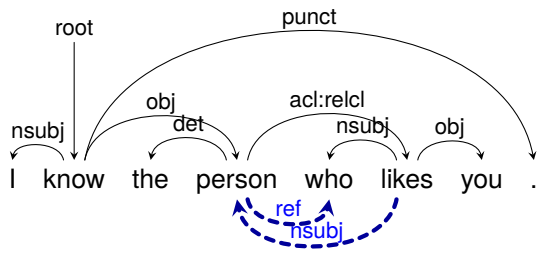
Keywords: AMR, Relative Clause, Semantic Parsing

1. Introduction

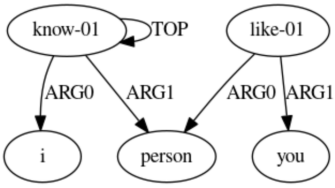
Abstract Meaning Representation (AMR; Banarescu et al., 2013) has emerged as a mainstream framework in semantic parsing tasks. Recent advancements in AMR parsers have led to significant achievements, with scores over 0.85 (Lee et al., 2022) in Smatch (Cai and Knight, 2013). However, relying solely on overall F-scores does not fully reveal a parser's performance across different linguistic phenomena, leaving areas for improvement and potential problems unclear.

In semantic parsing tasks, previous research has shown that sequence-to-sequence (seq2seq) models are good at abstracting away from surface variation in how meanings are expressed (Shaw et al., 2021). However, seq2seq models that process symbolic structures as mere strings face challenges in compositional generalization, such as the ability to process recursion, compared to models designed to be sensitive to the structure (Yao and Koller, 2022; Li et al., 2023; Shaw et al., 2021). This raises the possibility that such "structure-awareness" in the design of semantic parsers may be valuable for complex constructions generally.

In this paper, we focus on evaluating AMR parsers on English relative clauses, a frequent Long-Distance Dependency (LDD) construction. LDD refers to the linguistic phenomenon that two elements in a sentence, though not adjacent to each other, are still syntactically/semantically constrained. As a typical example of LDD, RCs are a popular topic of computational linguistic study (e.g., Davis and van Schijndel, 2020; Ravfogel et al., 2021). Compared with non-LDD constructions, RCs are structurally complex and may give rise to semantic ambiguities, so we assume they will be challenging for parsers. Figure 1 shows syntactic dependencies and the semantic AMR graph for an example RC.



(a) UD tree of the sentence: basic dependencies (above) and enhanced dependencies added for the RC (below).



(b) Normalized AMR graph. The `ARG0` edge from `like-01` to `person` corresponds to the relative clause.

(c) Canonical AMR graph. The `ARG0-of` edge corresponds to the relative clause.

Figure 1: UD and AMR representations for the sentence containing a subject relative clause *I know the person who likes you.* Converting the canonical (annotated/parsed) graph into the normalized one entails inverting the `-of` edges, causing nodes to be reentrant (have multiple parents).

In our evaluation we examine two types of AMR parsers: structure-aware and structure-unaware models. **Structure-unaware** models, as defined herein, process input purely as sequential strings; algorithms for learning and decoding are indifferent

5th Workshop on Designing Meaning Representations (DMR 2024) @LREC-COLING-2024, pages 151–161

to any notation within these strings that represents sentence structure. Conversely, **structure-aware** models are designed to take into account structural information, thereby enabling a more nuanced understanding of the input data's inherent syntactic and semantic properties.

We ask: **How well can AMR parsers capture the long-distance predicate-argument dependencies in RCs?** To answer this question, we normalize edges that contain `-of` by inverting the source node and the target node, and then evaluate parsers by measuring recall of the reentrancies introduced by RCs in two datasets: Universal Dependencies English Web Treebank (EWT; Silveira et al., 2014) and Controlled RCs (CRC; Prasad et al., 2019). Our investigation engages with the following subquestions:

- Does structure-awareness help the models to parse RCs in EWT and CRC?
- Which types of RC are most challenging and why?

Our contributions include:

- A fine-grained method to classify RCs and annotate Enhanced Universal Dependencies (EUD) in reduced RCs automatically.
- A systematic comparison of five AMR parsers, focusing specifically on their accuracy in parsing reentrancies introduced by RCs, along with an analysis of the underlying reasons for their performance differences.

This paper begins with an overview of RCs and reentrancies in AMR parsing (§2), followed by an introduction to the dataset, classification algorithm, and models in §3. §4 presents and discusses the results of our evaluation. The conclusion and suggestions for future research directions are presented in §5.[1]

2. Background & Related Work

2.1. Relative Clauses

In a canonical RC, a noun is modified by a clause and is understood to fulfill a grammatical function within that clause. The modified noun is the *head* of the RC. Some RCs have a *relative pronoun* like *which* or *that*. When the relative pronoun is omitted, the clause is termed a *reduced RC*; when the relative pronoun is present, along with a full clause structure, it is termed a *full RC*. According to the NP accessibility principle (Keenan and Comrie, 1977), English allows relativization on all grammatical functions. In the present study, we focus on four types of full RCs and two of their reduced counterparts:

- **Subject RC**: the relative pronoun functions as the subject of the active voice clause, as in: *He is the person* ***who stole my book***.
- **Object RC**: the relative pronoun functions as the object of the clause: *He is the person* ***that you like***.
- **Oblique RC**: the relative pronoun functions as an oblique within the RC: *He is the person* ***from whom I borrowed the book***. All PPs attaching to verbs/adjectives are considered obliques within the UD framework, which does not distinguish oblique arguments vs. adjuncts.
- **Passive RC**: the RC is a passive clause whose subject is relativized: *He is the person* ***who is accused of stealing my book***.
- **Reduced Object RC**: there is no relativizer but the head noun is understood to function as the object of the clause: *He is the person* ***you like***.
- **Reduced Oblique RC**: there is no relativizer but the head noun is understood to function as the oblique of the clause: *He is the person* ***I borrowed the book from***.

These are not the only kinds of RCs: there are also free relatives (e.g., *I heard what* ***you said***), possessive RCs (e.g., *I like the girl* ***whose dress is blue***), and reduced subject RCs (e.g., *I met the person* ***you mentioned __ finished all the work this week***; for clarity in this example, we indicate the site of the gap, i.e. where the noun would go were it not relativized).[2] However, as these are relatively rare in our dataset, our experiments are focused on the six major RC types listed above.

2.2. RCs in UD

For the present study, we use the framework of Universal Dependencies (UD, specifically UDv2; Nivre et al., 2020; de Marneffe et al., 2021), a syntactic annotation framework consisting of bilexical dependencies. UD defines a shallow dependency tree known as the *basic* tree, optionally complemented with an *enhanced* graph that adds deeper dependencies for several constructions.

The basic tree plus edges specific to the enhanced graph are illustrated in Figure 1a for a sentence with a subject RC. In the UD framework, (most) English RCs are considered a subtype of adnominal clause. The predicate of the RC attaches to the head noun with the `acl:relcl` dependency

[1] Our code and data can be found at https://github.com/xiulinyang/relative-amr-eval

[2] Adnominal participial clauses (*the sheep* ***eaten by wolves***, *the wolves* ***eating the sheep***) are considered RCs in some frameworks, but not in English UD (https://universaldependencies.org/en/dep/acl-relcl.html). There are also adverbial clauses analyzed as RCs in UD (`advcl:relcl`), e.g. in cleft sentences: *It was Booth* ***who shot Lincoln***. These are not very frequent in our data and we exclude them from our analysis.

relation. When a relative pronoun exists, in the basic tree, it attaches inside the RC with the relativized dependency relation. In the enhanced UD (EUD) representation, the head noun acquires the grammatical function within the RC, and the relative pronoun (if present) attaches to the head noun via a `ref` edge in lieu of its basic function.

2.3. Fine-grained AMR Evaluation

Recognizing that overall F-scores do not tell the full story of parser behavior, researchers have sought to provide a finer-grained picture of the performance of AMR parsers. Damonte et al. (2017) report the results of a wide range of general features of AMRs such as reentrancies, negative polarity, and wikification. To evaluate reentrancies, they normalize the edges in AMR so that RCs also introduce reentrancies. Our evaluation on AMR 3.0 data adopts their approach.

Szubert et al. (2020) provided a detailed analysis of reentrancies in AMR 2.0 caused by different syntactic, semantic, or pragmatic factors. They developed a set of heuristics to detect causes of reentrancies for parser evaluation. However, they focus on reentrancies in the canonical form of the AMR, whereas RCs are only reentrant in the inverse-normalized form (Figure 1), so they exclude RCs from their evaluation (Szubert et al., 2020, p. 2201).

The GrAPES benchmark (Groschwitz et al., 2023) is designed to test AMR parsers against nine specific challenging categories, which include structural generalization and syntactic as well as semantic reentrancies, among others. The dataset includes 130 RCs in a more challenging setting where sentences contain recursive RCs with optional coreference. Groschwitz et al. test three AMR parsers, all of which attain very low exact match scores ranging from 0% to 17% on these recursive RCs.

Our paper contributes to this literature by taking a deep dive on RCs, with an extensive comparison of AMR parsers across more than 1,400 corpus examples and 1,400 synthetic instances of RCs.

2.4. Probing Language Models using RCs

A few studies have used RCs to probe syntactic structures represented in language models (LMs) (e.g., Davis and van Schijndel, 2020; Mosbach et al., 2020; Prasad et al., 2019). They use either synthetic or naturalistic data to probe if the LM represents certain linguistic features or bias. For example, Davis and van Schijndel (2020) use English and Spanish RCs to examine the linguistic bias of RNN LM on the high/low attachment of RCs when trained with only synthetic or real multilingual corpus data. They found that models trained on synthetic data could learn to attach RCs both high and low in the sentence structure. However, when trained on real-world, multilingual corpus data, the models tended to favor low attachment, similar to the pattern seen in English, even though this preference is not common globally across languages. Following Kim et al. (2019); Warstadt and Bowman (2019), Mosbach et al. (2020) examined 3 pre-trained masked language models (BERT, RoBERTa, and ALBERT) on sentence-level syntactic and semantic understanding. They found that all models show high performance in parsing syntactic information but fail to predict the masked relative pronoun using context and semantic knowledge.

Dataset	# sents	# tokens
EWT	1,449	26.5
CRC	1,400	13.7
AMR 3.0	259	29.1

Table 1: Number of sentences containing RCs in the datasets and the mean sentence length

3. Method

3.1. Data

In our experiments, three datasets are used. The statistics of the dataset are reported in Table 1.

EWT UD treebank (henceforth EWT) The data we use is the train split from the Universal Dependencies English Web Treebank (Silveira et al., 2014). The original English Web Treebank contains constituency trees for diverse web text genres including weblogs, newsgroups, emails, reviews, and Yahoo! answers (Bies et al., 2012). It was then incorporated into the Universal Dependencies project; we use the dependency trees for this project.[3] Opting for the training split allows for a more extensive set of examples for evaluation. A thorough review has confirmed that there is no content overlap between EWT and the AMR 3.0 dataset (on which AMR parsers were trained).

Controlled RCs (henceforth CRC) The CRC dataset is adopted from (Prasad et al., 2019). It contains 7 types of clauses with controlled vocabulary and syntactic structures, which have been artificially generated to ensure balance across constructions and avoid potential confounds like length in comparing parser performance. We employed the four types of RCs in the dataset: subject RC, object RC, reduced object RC, and passive RC. Every category contains 350 examples.

[3] https://github.com/UniversalDependencies/UD_English-EWT/, specifically the dev branch as of Jan. 22, 2024, which contains changes beyond the UD 2.13 release

AMR 3.0 We report standard AMR parsing metrics on the test split of the AMR 3.0 release (Knight et al., 2021), which consists of gold AMR annotations from a variety of genres, including especially news and online discussion forums. We also report reentrancy recall on subject relative clauses.

3.2. RC classification

To have a fine-grained evaluation, we need to classify the sentences into different RC categories. We designed a straightforward algorithm to do this task. The classification results are then manually checked.

We first identify all sentences annotated with `acl:relcl`, totaling 2036 instances. Subsequently, these sentences were categorized based on the Enhanced Universal Dependency (EUD) relations attributed to the relativized head noun. Our six target subtypes are derived from the EUD relation and whether it is a full or reduced RC: `nsubj` (full), `obj` (full and reduced), `obl` (full and reduced), `nsubj:pass` (full). All other variations, such as possessives, were consolidated under the *Others* category as shown in Table 2. Please note that the total count in the table does not match 2036 due to sentences that contain multiple types of RCs.

Reduced RC classification Enhanced UD relations were present for full RCs (having been added based on the relativizer's dependency relation in the basic layer) but were missing for reduced RCs. To infer the enhanced relation in reduced RCs, we implement rules to identify the locally missing (gapped) function of the RC. For example, in *He is the person* ***you like*** __, in the basic UD tree the verb *like* has a subject dependent but no object, which is used to infer that it is a reduced object RC.

Our implementation takes into account the overall transitivity of the RC predicate verb (whether it tends to be transitive or intransitive). We combine data from a verb transitivity file[4] and the dependency relations of verbs found in EWT. Treebank information is given precedence; if relations like `xcomp` or `ccomp` are among the top three most frequent associations with a verb, we classify it as transitive. Otherwise, we rely on the transitivity data from our table.

Next, we extract the set of relation labels of dependents of the RC predicate, applying recursion for instances of `xcomp` and `ccomp` so as to handle sentences such as *After I have done all the work* ***I promised to do****, I will take a break.* We then look for a missing relation: For transitive or ditransitive verbs, we categorize the clause as a reduced object RC if the verb has no `obj` dependent, and as a reduced oblique RC otherwise. Clauses associated with intransitive verbs are invariably considered oblique RCs. The procedure produces 340 reduced object RCs and 218 reduced oblique RCs.[5] The detailed statistics can be found in Table 2.

RC Category	Count	%
Subject RC	725	35.3
Object RC	161	7.8
Oblique RC	139	6.8
Passive RC	100	4.9
Reduced object RC	340	16.5
Reduced oblique RC	218	10.6
Others	373	18.1
Total	2056	100.0

Table 2: Distribution of RC types in the EWT data we used for evaluation.

Our method relies heavily on the information about the transitivity of verbs. Each verb type is assumed to be either transitive or intransitive, which makes ambitransitive verbs a tricky case. For example, in the two NPs *the day* ***he returned*** and *the piece* ***he returned***, the first relativizes an adverbial adjunct, while the second one is an object relative. However, in our verb transitivity table, *return* is a transitive verb, so the first example is mistakenly tagged as a reduced object RC. Most of the classification errors are caused by this problem.

Another tricky case is embedded complement clauses or control/raising constructions that are marked with `ccomp` or `xcomp` in UD separately. Consider the following two sentences:

(1) I will do all the work **I need to do** __

(2) I will talk to all the people **I need** __ **to do the work**.

If we extract all the dependencies of the predicate verb *need*, we will get the same relations: `nsubj`, `xcomp`, `obj`. However, as we can see, the missing object is in different embedded structures and therefore, the enhanced UD relation will be wrong in terms of the head. We therefore collected all RCs with `xcomp`/`ccomp` for manual correction.

We manually checked and corrected all examples in each reduced RC type. The results demonstrate high accuracy in discriminating the two classes, with a recall of 94% for reduced object RCs and 95% for reduced oblique RCs.

[4] https://github.com/wilcoxeg/verb_transitivity The CSV file contains the percentage of the time the verb is transitive, intransitive, and ditransitive in the Google syntactic ngrams corpus.

[5] Note that reduced subject RCs only occur in doubly embedded clauses (e.g. *the rooster* ***I thought was a hen***). These are rare and were dealt with manually.

AMR 3.0	All Sentences		RC Sentences		
Models	F (Full graph)	F (All reentrancies)	F (Full graph)	F (All reentrancies)	Subj RC Recall
AM-Parser[§]	74.9	57.0	73.5	57.7	65.2
amrlib-T5	82.0	71.4	77.6	70.4	71.0
amrlib-BART	82.3	73.5	80.6	73.3	**79.0**
Spring	83.0	68.0	72.5	65.5	65.2
AMRBART[§]	**84.2**	**74.3**	**80.8**	**73.4**	75.4

Table 3: Smatch F_1 scores and subject RC reentrancy recall of the models on AMR 3.0 test split. Two kinds of F_1 scores are shown: overall Smatch score comparing the full graph to the gold standard AMR, and the Reentrancies subscore (Damonte et al., 2017). These are shown for the full test set as well as the subset of test sentences containing a relative clause. The last column shows recall of reentrancies on subject relative clauses (138 examples in total; other RC subtypes were less frequent). "§" superscript means "structure-aware". The first four measures do not require token-level alignments between the graph and the text.

3.3. Models

In our experiments, we test five different models. The first, AM-Parser, derives a parse compositionally after predicting supertags and dependencies. The other four are sequence-to-sequence models, one of which has a structure-aware component in its training loss.

Structure-aware models AM-Parser (Groschwitz et al., 2018) is a neuro-symbolic compositional semantic parser that learns the sub-graphs of meaningful tokens and then combines them for a complete AMR. It is trained on two objectives: (a) learning the supertags aligned with each token; and (b) learning the dependency trees that connect the supertags to build a complete AMR graph. The supertagger and dependency parser are both trained on bert-large-uncased model.

AMRBART (Bai et al., 2022) is a graph-pretrained model based on BART (Lewis et al., 2020). Unlike traditional text-only pretraining, AMRBART masks parts of AMR graphs—like nodes and edges—during pretraining. It introduces a unified pretraining framework that combines the original text with its AMR graph, ensuring the model learns both linguistic content and graph structure. For pretraining, it uses 20k silver-standard AMR graphs created by Spring (Spring et al., 2021), and then it is fine-tuned with gold AMR data. The fine-tuned model shows more robust performance on unseen data, highlighting its potential for complex language tasks that require deep understanding.

Structure-unaware models We examined three structure-unaware models. They are pretrained language models fine-tuned on linearized AMRs with necessary preprocessing.

Spring (Spring et al., 2021) fine-tunes BART-base with vocabulary expansion. To achieve better results, instead of using linearized PENMAN notation, they adopt graph linearization by replacing variables with special tokens <Rx> where x is a number. In this way, the constants and variables in AMRs can be distinguished. Despite the preprocessing steps, the model still takes the input as sequence of strings without distinguishing the structural information and hence we categorize Spring as a structure-unaware model.

Similarly, amrlib fine-tunes the pre-trained language models such as BART-large and T5 models to translate natural language to linearized AMR.[6]

3.4. Evaluation

Our evaluation assesses whether the relativized noun in a sentence is reentrant, with two incoming edges—one originating from the main clause's predicate verb and another from the predicate within the RC. Take the sentence in Figure 1 as an example. After normalizing all the inverse edges, our script identifies the RC from the acl:relcl edge going from *person* to *likes*. It identifies the associated AMR nodes, person and like-01, and checks whether (1) the person node receives two incoming edges, and (2) there is an edge from like-01 to person. If so, the reentrancy expected for the RC is scored as recovered by the parser.

This analysis requires alignments between tokens in the sentence and their semantic nodes in order to determine, given a relative clause predicate p and its head noun n, which AMR edge (if any) is the associated reentrancy of the form $p \rightarrow n$. For AM-Parser, which inherently requires node-token alignment, we extract these alignments directly from its predictions. For the other parsers under study, we utilize LEAMR (Blodgett and Schneider, 2021), a probabilistic, fine-grained aligner optimized for English AMR.

Our evaluation metric is the **recall** in counting instances where the head noun's aligned node receives edges from both the main and RC predicate nodes. This approach allows us to effectively gauge

[6] https://github.com/bjascob/amrlib/wiki/The-parse_xfm-model

Model	Subj RC	Obj RC	Pass RC	Obl RC	RedObj RC	RedObl RC	All
AM-Parser§	57.4 (416/725)	55.3 (89/161)	74.0 (74/100)	33.3 (46/138)	50.6 (172/340)	34.4 (75/218)	51.8
	83.4 (605/725)	84.4 (136/161)	84.0 (84/100)	78.2 (108/138)	86.5 (294/340)	70.6 (154/218)	82.1
amrlib-BART	67.7 (491/725)	64.0 (103/161)	**80.0 (80/100)**	**65.2 (90/138)**	**62.1 (211/340)**	45.0 (98/218)	**63.8**
	87.2 (632/725)	83.9 (135/161)	94.0 (94/100)	87.0 (120/138)	80.6 (274/340)	67.0 (146/218)	83.2
amrlib-T5	**68.0 (493/725)**	**67.1 (108/161)**	77.0 (77/100)	55.1 (76/138)	59.4 (202/340)	45.4 (99/218)	62.7
	85.9 (623/725)	85.7 (138/161)	97.0 (97/100)	81.9 (113/138)	80.0 (272/340)	67.4 (147/218)	82.6
Spring	63.6 (461/725)	58.4 (94/161)	79.0 (79/100)	57.2 (79/138)	52.4 (178/340)	38.1 (83/218)	57.9
	81.5 (591/725)	76.4 (123/161)	94.0 (94/100)	76.8 (106/138)	73.5 (250/340)	56.4 (123/218)	76.5
AMRBART§	65.7 (476/725)	62.1 (100/161)	**80.0 (80/100)**	**65.2 (90/138)**	58.8 (200/340)	46.8 (102/218)	62.3
	85.5 (620/725)	80.1 (129/161)	94.0 (94/100)	87.0 (120/138)	79.1 (269/340)	69.7 (152/218)	82.3
Average	64.5 (467/725)	61.2 (99/161)	78.0 (78/100)	55.2 (76/138)	56.6 (193/340)	41.9 (91/218)	59.1

Table 4: Results by parser and RC type on the EWT dataset. Structure-aware parsers are notated with §. White rows report recall of RC-triggered reentrancy edges. Gray rows report *attainability* rates subject to the predicted nodes and their token alignments; this is an upper bound of recall. 3 graphs produced by AMRBART cannot be aligned with LEAMR, so we remove them from the evaluation set. The best results in each column and condition are indicated in **bold**.

Model	Subj RC	Obj RC	Passive RC	RedObj RC	All
AM-Parser§	96.0 (335/349)	96.0 (332/346)	97.1 (340/350)	92.4 (280/303)	95.5 (1,287/1,348)
amrlib-BART	**98.6 (344/349)**	98.0 (339/346)	**99.1 (347/350)**	**98.3 (298/303)**	**98.5 (1,328/1,348**)
amrlib-T5	98.3 (343/349)	97.7 (338/346)	98.9 (346/350)	94.0 (284/303)	97.3 (1,311/1,348)
Spring	97.7 (341/349)	**98.3 (340/346)**	**99.1 (347/350)**	98.0 (297/303)	98.3 (1,325/1,348)
AMRBART§	97.4 (340/349)	96.8 (335/346)	98.9 (346/350)	97.4 (295/303)	97.6 (1,316/1,348)
Average	97.6 (341/349)	97.3 (338/346)	98.6 (345/350)	96.0 (291/303)	98.0 (1,321/1,348)

Table 5: Recall by parser and RC type on the CRC dataset of synthetic sentences.

the parsers' proficiency in handling reentrancies within the constraints of available data.[7]

4. Results & Discussion

4.1. Overall Results

The initial assessment of the models was conducted on the AMR 3.0 test split (after running a dependency parser to find RCs), with outcomes presented in Table 3. The findings indicate that overall seq2seq models show better performance than the compositional AM-Parser model. Across metrics, AMRBART and amrlib-BART show good performance relative to other models.

It is also noteworthy that in the parsing of sentences with RCs, all models exhibit a decline in F-score, with Spring experiencing a sizable drop (from 83.0 to 72.5). This decrease may be attributed to the long-distance dependencies and more complex syntactic structures that relative clauses introduce.

The accuracy of 5 different models in processing various RC types in the new datasets is systematically examined and reported in Tables 4 and 5 for corpora with gold syntax annotations. If the predicate token or head token is not aligned to a node, it is impossible to get the reentrancy. Therefore, we also report the **attainability rate**, the rate at which node-token alignments could be recovered for both the RC head and predicate tokens, as seen in the gray rows of Table 4. If an RC reentrancy is *un*attainable, it means either that one or both of its tokens lack a corresponding node in the predicted AMR (usually a parser error), or that it was present but could not be aligned in post-processing (for systems where this step was necessary, namely the seq2seq models).

[7] We do not evaluate the role label on the reentrancy edge, because the role numbers in AMR predicates (mostly sourced from PropBank; Kingsbury and Palmer, 2002; Pradhan et al., 2022) are semantic rather than syntactic, and thus will not line up perfectly with the syntactic RC categories. However, the numbering conventions are weakly connected to syntactic functions: we expect that ARG0 should imply a subject RC; a subject RC should imply ARG0 or ARG1; a passive RC should usually imply ARG1; an object RC should imply ARG1 or ARG2; and ARG3, ARG4, etc. should generally imply an oblique RC (full or reduced). Non-core roles would likely correspond to obliques as well.

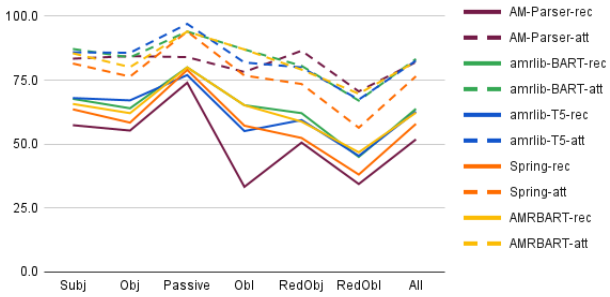


Figure 2: RC reentrancy recall (solid lines) and attainability rate (dashed) of all parsers, by RC subtype and overall.

```
(p0 / serve-01
    :ARG0 (p1 / person
              :wiki "George W. Bush"
              :name (p2 / name
                        :op1 "Bush"))
    :duration (p3 / nearly
                  :op1 (p4 / temporal-quantity
                            :quant 2
                            :unit (p5 / year)))
    :time (p6 / over-01
              :ARG1 (p7 / it)))
```

Figure 3: Predicted AMR for the sentence *By the time it was over, Bush had served nearly two years.*

EWT Overall, as reported in Table 4 and visualized in Figure 2, detecting the edges between the relative predicate and the head noun is challenging for all models, with recall below 64%.[8] This suggests that relative clause structures are especially difficult.

Comparison of parsers. Our results reveal that seq2seq parsers, whether they are structure-aware or not, outperform the compositional `AM-Parser`. Moreover, the overall performance of all seq2seq models is very similar. The performance of `AM-Parser` in parsing RCs appears less advantageous, which we conjecture may stem from the pretrained language model (i.e. `bert-large-uncased`) used. As we can see, the two `BART`-based parsers perform the best. Further exploration of the role of pretrained language models is left to future work.

Attainability. According to Table 4, we can see that even when both the head token and predicate token have predicted nodes, there remains considerable scope for further improvement given that UD parsing has reached over 95% in LAS since 2018 (e.g., Clark et al., 2018). This means that structural information is not fully captured by all models.

However, we recognize that the low recall might stem from the alignment model utilized. The attainability rate for oblique reduced RCs is particularly low, which likely affects recall scores. Misalignments between some subgraphs and tokens are observed; since our analysis targets subgraphs aligning with both the head and predicate tokens, such misalignments can diminish the scores. Additionally, it is possible that tokens are classified as edges rather than nodes, as illustrated in Figure 3 where no node but just an edge is aligned with the token *time*.

RC subtypes. Oblique, reduced oblique, and reduced object RCs are particularly hard. Psycholinguistic research has shown that oblique relative clauses are more challenging for humans to process due to the greater distance between the filler and the gap, compared to other types of relative clauses (e.g., Diessel and Tomasello, 2005); this distance may also be challenging for the AMR parsers. That reduced RCs are harder to parse than full RCs is likely due to the lack of explicit syntactic cues. It is interesting to see that passive RCs are easiest to parse of the RC categories. This is probably because both the relative pronoun and the passive construction provide more linguistic cues than other types of RCs. Subject RCs, the most frequent category in both the EWT and AMR 3.0 datasets (especially if the passive subjects are included), are easier than non-subject RCs. Psycholinguistic studies have shown subject RCs to be easier for humans to comprehend and acquire (Gordon and Lowder, 2012; Diessel and Tomasello, 2005), and Reali and Christiansen (2007) found that more frequent RC types are easier to process (but did not consider passive subject RCs).

CRC As for the synthetic data, scores are quite high across parsers and RC categories. `amrlib-BART` marginally outperforms other models on average. For object-reduced RCs, `AM-Parser` and `amrlib-T5` are notably weaker than the other systems. The CRC dataset does not contain any oblique RCs, so there is no relevant result on this category. The results for parsing different types of RCs presented in Table 5 align closely with those reported in Table 4.

4.2. Exploring Parsing Performance Variations in RCs

The models vary in absolute scores, but they follow a general trend: reentrancies in passive RCs are more often recovered than those in subject RCs, followed by object RCs and oblique RCs. Reduced RCs are harder to predict. We observe a similar pattern in the CRC data both in dependency and semantic parsing. Next we explore two possible factors influencing parsing performance across RCs, namely, dependency distance and training data distribution.

[8] For the `amrlib-BART` model (overall recall of 63.8%), we also computed recall of AMR edges for `ccomp` complement clauses, which was much higher: 77.4% (1445/1868), with an attainability rate of 82.7 (1545/1868).

RC Category	Dep Dist	Mean Recall
Reduced oblique RC	3.06	41.9
Reduced object RC	3.13	56.6
Subject RC	4.30	64.5
Passive RC	5.78	78.0
Object RC	5.21	61.2
Oblique RC	6.98	55.2

Table 6: Mean dependency distance of 6 types of RCs in our experiments

RC Category	Count
Subject RC	4,226
Object RC	516
Oblique RC	729
Passive RC	534
Reduced object RC	1,371
Reduced oblique RC	1,092

Table 7: Distribution of 6 RC types in AMR 3.0 train split

Dependency distance Dependency distance refers to the linear distance between two words connected by a dependency relation, which functions as an important indicator of syntactic difficulty (Liu et al., 2017). Existing research has reported that longer dependency distance makes subject RCs easier to process than object RCs in English (Gibson, 1998) and vice versa in Chinese (Hsiao and Gibson, 2003). In this paper, we calculate the mean dependency distance between the predicate in the RC and the head noun in the matrix clause in each type of RC. It is surprising that the reduced RCs have the shortest dependency distance even if we assume the existence of the relative pronoun (i.e., we add 1 to the existing dependency distance). The shorter distance might justify dependency distance minimalization (Temperley, 2007) because the omission of the relative pronoun makes the sentence harder to process and therefore only shorter dependency distance makes them easier to process.

Regarding the full RCs, as shown in Table 6, in the EWT dataset, the dependency distance largely meets the observation made by previous research that subject RC is easier than object RC. Notably, passive RCs, despite their longer dependency distances, exhibit high parsing accuracy. This could be attributed to passive RCs essentially acting as subject RCs, with the relative pronoun serving as the subject. When considering subject and passive RCs together, the average dependency distance decreases to 4.46, making these types the most straightforward for parsers.

Training data distribution We investigated the distribution of different RC types within the AMR 3.0 training split. Given the absence of gold-standard dependency annotations in AMR 3.0, we obtained automatic dependency trees using `Stanza`.[9] For full RCs, classification was based on the dependency relationship between the relative pronoun and its predicate. The identification of reduced RCs employed the methodology outlined in §3.2. As Table 7 illustrates, the prevalence of RC types in AMR 3.0 closely mirrors that of EWT, with subject RCs being the most common.

[9] `stanza-1.6.0`: https://github.com/stanfordnlp/stanza/releases/tag/v1.6.0

It is intriguing that despite being more common, subject RCs are still tougher to handle than their passive forms. This revelation suggests that the frequency of a structure does not necessarily make it easier to process, hinting at deeper complexities in understanding syntactic patterns.

5. Conclusion

In our study, we compared two structure-aware AMR parsers (`AM-Parser` and `AMRBART`) and typical structure-unaware seq2seq models (`Spring`, `amrlib-BART`, and `amrlib-T5`) in parsing relative clauses. We find that relative clauses are challenging for current parsers. Seq2seq models, on the whole, outperform the compositional model. Interestingly, there is little difference in performance between seq2seq models that are aware of structure and those that are not. Furthermore, our analysis reveals that (reduced or full) oblique and reduced object RCs are the most challenging RC types. Examining the relationship to dependency length, we find that the full RCs with shorter dependency distances are easier to parse; however, reduced RCs with the shortest dependency distance are more challenging for all parsers. As part of our study, we have produced gold EUD annotations for reduced RCs in the English Web Treebank; these will be released upon publication.

Future work might expand the scope of inquiry to more diverse reentrancy types by leveraging the (E)UD annotations. It would also be interesting to see if adding (E)UD information to AMR parsing helps the structure-unaware parsers to learn the complex structural information (cf. Findlay and Haug, 2021).

Acknowledgments

We thank Jonas Groschwitz and Austin Blodgett for their invaluable technical assistance, and members of the NERT lab for their insightful feedback. We are also grateful to the anonymous reviewers whose suggestions have significantly enhanced this paper. This research was supported in part by NSF award IIS-2144881.

Bibliographical References

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for AMR parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6001–6015, Dublin, Ireland. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English Web Treebank. LDC2012T13.

Austin Blodgett and Nathan Schneider. 2021. Probabilistic, structure-aware algorithms for improved variety, accuracy, and coverage of AMR alignments. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3310–3321, Online. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Brussels, Belgium. Association for Computational Linguistics.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for Abstract Meaning Representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain. Association for Computational Linguistics.

Forrest Davis and Marten van Schijndel. 2020. Recurrent neural network language models always learn English-like relative clause attachment. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1979–1990, Online. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Holger Diessel and Michael Tomasello. 2005. A new look at the acquisition of relative clauses. *Language*, pages 882–906.

Jamie Y. Findlay and Dag T. T. Haug. 2021. How useful are enhanced Universal Dependencies for semantic interpretation? In *Proceedings of the Sixth International Conference on Dependency Linguistics (Depling, SyntaxFest 2021)*, pages 22–34, Sofia, Bulgaria. Association for Computational Linguistics.

Edward Gibson. 1998. Linguistic complexity: Locality of syntactic dependencies. *Cognition*, 68(1):1–76.

Peter C Gordon and Matthew W Lowder. 2012. Complex sentence processing: A review of theoretical perspectives on the comprehension of relative clauses. *Language and Linguistics Compass*, 6(7):403–415.

Jonas Groschwitz, Shay Cohen, Lucia Donatelli, and Meaghan Fowlie. 2023. AMR parsing is far from solved: GrAPES, the granular AMR parsing evaluation suite. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10728–10752, Singapore. Association for Computational Linguistics.

Jonas Groschwitz, Matthias Lindemann, Meaghan Fowlie, Mark Johnson, and Alexander Koller. 2018. AMR dependency parsing with a typed semantic algebra. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1831–1841, Melbourne, Australia. Association for Computational Linguistics.

Franny Hsiao and Edward Gibson. 2003. Processing relative clauses in Chinese. *Cognition*, 90(1):3–27.

Edward L. Keenan and Bernard Comrie. 1977. Noun phrase accessibility and universal grammar. *Linguistic Inquiry*, 8(1):63–99.

Najoung Kim, Roma Patel, Adam Poliak, Patrick Xia, Alex Wang, Tom McCoy, Ian Tenney,

Alexis Ross, Tal Linzen, Benjamin Van Durme, Samuel R. Bowman, and Ellie Pavlick. 2019. Probing what different NLP tasks teach machines about function word comprehension. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 235–249, Minneapolis, Minnesota. Association for Computational Linguistics.

Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proc. of LREC*, pages 1989–1993, Las Palmas, Canary Islands.

Kevin Knight, Bianca Badarau, Laura Baranescu, Claire Bonial, Madalina Bardocz, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, Tim O'Gorman, and Nathan Schneider. 2021. Abstract Meaning Representation (AMR) Annotation Release 3.0. Linguistic Data Consortium, LDC2020T02.

Young-Suk Lee, Ramón Astudillo, Hoang Thanh Lam, Tahira Naseem, Radu Florian, and Salim Roukos. 2022. Maximum Bayes Smatch ensemble distillation for AMR parsing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5379–5392, Seattle, United States. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Bingzhi Li, Lucia Donatelli, Alexander Koller, Tal Linzen, Yuekun Yao, and Najoung Kim. 2023. SLOG: A structural generalization benchmark for semantic parsing. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3213–3232, Singapore. Association for Computational Linguistics.

Haitao Liu, Chunshan Xu, and Junying Liang. 2017. Dependency distance: A new perspective on syntactic patterns in natural languages. *Physics of life reviews*, 21:171–193.

Marius Mosbach, Stefania Degaetano-Ortlieb, Marie-Pauline Krielke, Badr M. Abdullah, and Dietrich Klakow. 2020. A closer look at linguistic knowledge in masked language models: The case of relative clauses in American English. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 771–787, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proc. of LREC*, pages 4027–4036, Marseille, France.

Sameer Pradhan, Julia Bonn, Skatje Myers, Kathryn Conger, Tim O'Gorman, James Gung, Kristin Wright-Bettner, and Martha Palmer. 2022. PropBank comes of age—larger, smarter, and more diverse. In *Proc. of *SEM*, pages 278–288, Seattle, Washington.

Grusha Prasad, Marten van Schijndel, and Tal Linzen. 2019. Using priming to uncover the organization of syntactic representations in neural language models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

Shauli Ravfogel, Grusha Prasad, Tal Linzen, and Yoav Goldberg. 2021. Counterfactual interventions reveal the causal effect of relative clause representations on agreement prediction. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 194–209, Online. Association for Computational Linguistics.

Florencia Reali and Morten H Christiansen. 2007. Processing of relative clauses is made easier by frequency of occurrence. *Journal of memory and language*, 57(1):1–23.

Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2021. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 922–938, Online. Association for Computational Linguistics.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel R. Bowman, Miriam Connor, John Bauer, and Christopher D. Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 2897–2904, Reykjavík, Iceland.

Nicolas Spring, Annette Rios, and Sarah Ebling. 2021. Exploring German multi-level text simplification. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1339–1349, Held Online. INCOMA Ltd.

Ida Szubert, Marco Damonte, Shay B. Cohen, and Mark Steedman. 2020. The role of reentrancies in Abstract Meaning Representation parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2198–2207, Online. Association for Computational Linguistics.

David Temperley. 2007. Minimization of dependency length in written English. *Cognition*, 105(2):300–333.

Alex Warstadt and Samuel R Bowman. 2019. Linguistic analysis of pretrained sentence encoders with acceptability judgments. *arXiv preprint arXiv:1901.03438*.

Yuekun Yao and Alexander Koller. 2022. Structural generalization is hard for sequence-to-sequence models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5048–5062, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.