Language Resources and Evaluation (2022) 56:1075–1102
https://doi.org/10.1007/s10579-022-09591-7

ORIGINAL PAPER

A tale of four parsers: methodological reflections on diagnostic evaluation and in-depth error analysis for meaning representation parsing

Maja Buljan[1] · Joakim Nivre[2,3] · Stephan Oepen[1] · Lilja Øvrelid[1]

Accepted: 5 April 2022 / Published online: 17 May 2022

Abstract
We discuss methodological choices in diagnostic evaluation and error analysis in meaning representation parsing (MRP), i.e. mapping from natural language utterances to graph-based encodings of semantic structure. We expand on a pilot quantitative study in contrastive diagnostic evaluation, inspired by earlier work in syntactic dependency parsing, and propose a novel methodology for qualitative error analysis. This two-pronged study is performed using a selection of submissions, data, and evaluation tools featured in the 2019 shared task on MRP. Our aim is to devise methods for identifying strengths and weaknesses in different broad families of parsing techniques, as well as investigating the relations between specific parsing approaches, different meaning representation frameworks, and individual linguistic phenomena—by identifying and comparing common error patterns. Our preliminary empirical results suggest that the proposed methodologies can be meaningfully applied to parsing into graph-structured target representations, as a side-effect uncovering hitherto unknown properties of the different systems that can inform future development and cross-fertilization across approaches.

Keywords Data-driven parsing · Sentence semantics · Meaning representation parsing · Contrastive evaluation · Diagnostics

1 Introduction

In the past decade, a branch of semantic parsing now commonly dubbed *meaning representation parsing* (MRP) has seen growing interest. Kate and Wong (2010) originally proposed an interpretation of *semantic parsing* as "the task of mapping

✉ Maja Buljan
majabu@ifi.uio.no

[1] Department of Informatics, Language Technology Group, University of Oslo, Oslo, Norway

[2] Department of Linguistics and Philology, Uppsala University, Uppsala, Sweden

[3] RISE Research Institutes of Sweden, Kista, Sweden

natural language sentences into complete formal meaning representations which a computer can execute for some domain-specific application." In contrast, MRP is characterized by domain- and task-independent *general-purpose* representations of sentence meaning in the form of *labeled directed graphs*. These graphs are linguistically interpretable and—albeit to variable degrees—reflect logic-based approaches to computational semantics that can facilitate formal reasoning. Standardised benchmarking data, evaluation metrics, and experimental results are available from a series of annual parsing competitions at the 2014 and 2015 workshops on Semantic Evaluation (Oepen et al., 2014, 2015) and the 2019 and 2020 Conference on Computational Natural Language Learning (CoNLL; Oepen et al., 2019, 2020).

Unlike most representations of syntactic structure, which limit themselves to *rooted trees*, common meaning representation frameworks assume general graphs. These structures make the parsing task much more complex—often moving from techniques with polynomial worst-case complexity to problems that are in principle NP-hard. Among other things, meaning representations transcend syntactic trees in allowing nodes with in-degree greater than one ("reentrancies"), multiple root nodes, and fewer constraints on the relation between elements of the graph and corresponding sub-strings of the parser input. Besides greatly increased modelling and algorithmic complexity, MRP also poses its own set of methodological challenges for parser evaluation, diagnostics, and error analysis.

For syntactic dependency parsing, the contrastive studies by McDonald and Nivre (2007, 2011) have been influential in comparing the performance of two core types of approaches, i.e. different families of parsing architectures. In this work, we investigate to what degree these techniques can be transferred to MRP, and how they can be adapted and extended to reflect the formal and linguistic differences in the nature of the target representations. We develop a general framework for quantitative diagnostic evaluation and experimentally validate this methodology through the application to four distinct parsers and three different meaning representation frameworks. Combining fine-grained quantitative and in-depth qualitative analysis, we propose a pipeline for error analysis, to inform both the comparison between parsers and across frameworks.

This work is an extension of the pilot study presented in Buljan et al. (2020). We generalise their techniques to an additional, formally very different framework, and to the output of an additional parsing system. Furthermore, we introduce another analytical methodology, moving from aggregate statistics to in-depth error analysis, with the ability to generalise and identify error patterns across parsing systems.

The remainder of the paper is structured as follows: In Sect. 2, we present the relevant background, including previous studies in syntactic parsing that provide our point of departure and the 2019 and 2020 shared tasks on MRP. Section 3 gives an overview of the experimental data used for analysis, and the parsing systems in focus. In Sect. 4, we present a sample multi-dimensional study on system performance depending on target framework, different types of graph elements, and parser input complexity. Section 5 introduces the in-depth, datapoint-oriented error analysis portion of our methodology. Finally, Sect. 6 reflects on our methodological proposals and empirical observations, and discusses avenues for future research.

2 Background

The following paragraphs establish relevant methodological and technological context for our work, out of necessity summarising prior efforts in rather broad strokes. For additional historic background, please see the discussion in Buljan et al. (2020).

2.1 MRP 2019 and 2020

The 2019 and 2020 instances of the Conference for Computational Language Learning (CoNLL) hosted a shared task series devoted to MRP across frameworks (Oepen et al., 2019, 2020). For the first time, these tasks have combined *formally* and *linguistically* different approaches to meaning representation in graph form in a uniform training and evaluation setup. The MRP training and evaluation data comprised a range of distinct approaches—which all encode core predicate–argument structure, among other things—to the representation of sentence meaning in the form of directed graphs, packaged in a unified abstract graph model and common serialisation format. In a nutshell, the MRP graph model defines four types of decorations on nodes: (a) an atomic node label, (b) a set of property–value pairs, (c) anchoring into the surface string (see below), and (d) a boolean indicator of the top node(s); correspondingly, graph edges can be decorated with: (e) an atomic label, (f) attribute–value pairs, and (g) anchoring. All seven types are optional, and meaning representation frameworks differ in which they take into use; in the present study, we will only encounter types (a) through (e).

The MRP task design has sought to enable cross-framework benchmarking of different parsing approaches and to advance learning from complementary knowledge sources (e.g. via parameter sharing). At the same time, the experimental results from these shared tasks enable diagnostic and contrastive evaluation across different types of meaning representation frameworks and across distinct parsing architectures. The MRP 2019 competition received submissions from eighteen teams, and the 2020 follow-up task by eight teams (many of them from among the top performers in the first edition).

In total, no less than seven linguistically distinct frameworks for meaning representation were included in the MRP task series, mostly for English but in 2020 also for Chinese, Czech, and German. For our empirical studies, we select from this set three of the more widely adopted frameworks, chosen to exemplify the extreme points along the dimension called *anchoring* in the MRP context, which characterizes the nature of the relation holding between graph elements and input sub-strings.

Regarding the most constrained form of anchoring, Fig. 1 shows two example graphs for one sentence from the venerable Wall Street Journal (WSJ) corpus in the two bi-lexical MRP frameworks, DELPH-IN MRS Bi-Lexical Dependencies (DM) of Ivanova et al. (2012) and Oepen and Lønning (2006), and Prague Semantic Dependencies (PSD) by Hajič et al. (2012) and Miyao et al. (2014). The DM and PSD frameworks are *bi-lexical* in the MRP collection, characterised by an injective relation between graph nodes and surface lexical units (tokens). In such graphs, each node is directly linked to a specific token (conversely, there may be

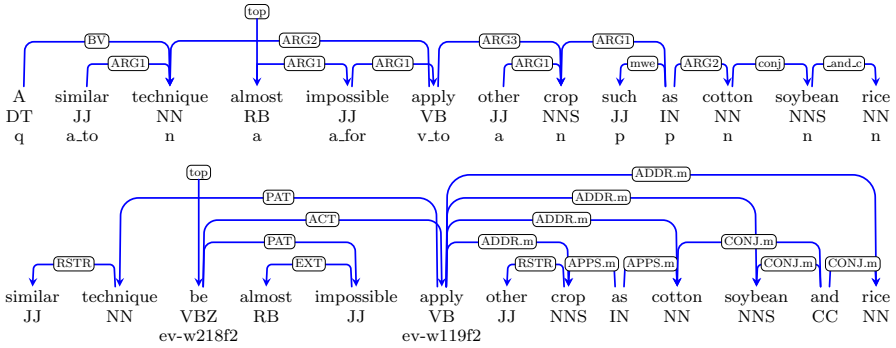

Fig. 1 Sample bi-lexical semantic dependency graphs for the example sentence *A similar technique is almost impossible to apply to other crops such as cotton, soybeans, and rice*. The top graph shows DELPHIN MRS Bi-Lexical Dependencies (DM), and the bottom one Prague Semantic Dependencies (PSD). In this linearized rendering of the graphs, the top row of node-local information in each graph shows the label (lemmas, in the case of DM and PSD), and the following two rows indicate two node properties: parts of speech (present on all nodes), and frame or sense identifiers (present only on verbal nodes in PSD)

semantically empty tokens), and there is a total ordering of nodes reflecting the linear order of their corresponding tokens. But even within this limiting assumption, which makes these graphs formally somewhat similar to standard syntactic dependency trees, the examples in Fig. 1 exhibit all the non-tree properties sketched in Sect. 1 above (reentrancies, multiple roots, and semantically vacuous surface tokens). DM and PSD nodes are labeled with lemmas and further decorated with two node properties: parts of speech, and (for verbs only, in the PSD case) frame or sense identifiers; jointly, these three components characterize a semantic predicate. Edges represent semantic argument roles: DM mostly uses overtly order-coded labels, e.g. ARG1, ARG2, etc. Abstractly similar, PSD labels like ACT (or), PAT(ient), or ADDR(essee) indicate 'participant' positions in an underlying valency frame.

As regards different anchoring relations, on the opposite end of the range of frameworks in the MRP shared tasks is Abstract Meaning Representation [AMR; Banarescu et al. (2013)], which by design does not spell out how nodes relate to sub-strings of the underlying parser input; Fig. 2 shows the same example sentence in AMR. While AMR edge labels resemble those of DM (essentially encoding predicate-specific argument positions rather than general thematic roles), the nodes in Fig. 2 are labeled with what AMR calls concept identifiers rather than words. AMR graphs are formally unordered and decline to make explicit how the different graph elements correspond to parts of the parser input string.

Without an explicit relation to the surface string, diagnostic evaluation involving AMR cannot invoke string-level characteristics (e.g. input length) or morpho-syntactic properties like parts of speech. Thus, several of the 'querying' dimensions from the study by McDonald and Nivre (2011) need to either be derived or replaced by other structural properties; we return to this question in Sect. 4 below.

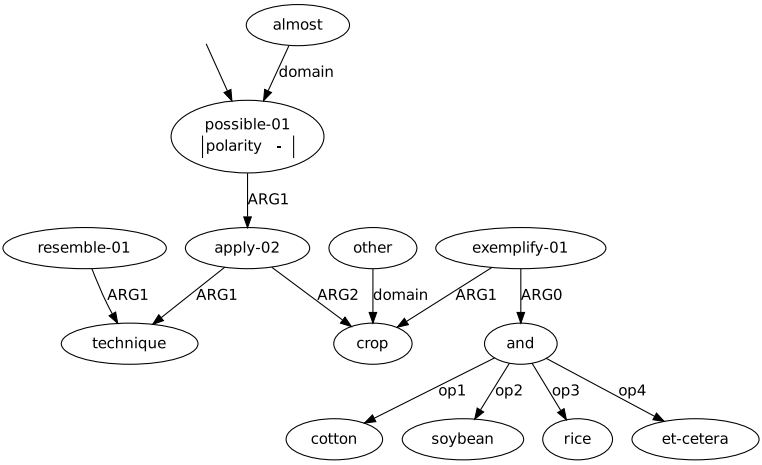

Fig. 2 Sample Abstract Meaning Representation (AMR) graph for the same sentence as in Fig. 1. Unlike the bi-lexical semantic dependencies, the graph is unordered and unanchored; all nodes and edges carry a label, and one node has an additional polarity property; the incoming arrow on that same node further signifies that it is the top of the graph

2.2 Cross-framework parser evaluation

Taking advantage of the uniform graph model across formally and linguistically different meaning representation frameworks, the MRP shared tasks develop a framework-independent metric to quantify parser success in terms of graph similarity between the gold-standard target graph and the actual parser output, generalizing closely related earlier work by Cai and Knight (2013), Damonte et al. (2017), Dridan and Oepen (2011); inter alios. The MRP evaluation metric is defined in terms of F_1 scores at the level of different *types* of individual graph elements (see above), e.g. node labels, additional node-local properties, identification of the top node(s), individual labeled edges, and (where applicable) the anchoring relation itself (edge attributes and edge anchoring are not present in our selection of frameworks from the MRP range of meaning representations). The top node and labeled edge components of the MRP metric closely correspond to established evaluation practices in syntactic dependency parsing, essentially scoring isolated dependency edges. However, the sub-problems of node identification, labelling, and anchoring take a much more prominent role in MRP (even for the bi-lexical MRP graphs), and some of our reflections below explicitly seek to tease apart parser behavior on node-local vs. more structural predictions.

2.3 Related work

The first part of the present study transfers the contrastive error analysis of *graph-based* vs. *transition-based* syntactic dependency parsers by McDonald and Nivre (2007, 2011) to MRP. Using multilingual data from the CoNLL 2006 shared task on dependency parsing Buchholz and Marsi (2006), the original study analysed the performance of the two parser types in relation to a number of structural factors, such as

sentence length, dependency length, and tree depth, as well as linguistic categories, notably parts of speech and dependency types. The analysis showed that, although the best graph-based and transition-based syntactic dependency parsers at the time achieved very similar accuracy on average, they had quite distinctive error profiles. More recently, Kulmizev et al. (2019) replicated this analysis for neural graph-based and transition-based syntactic dependency parsers, showing that—although the distinct error profiles are still discernible—the differences are now much smaller and are further reduced by the use of deep contextualised word embeddings Devlin et al. (2019), Peters et al. (2018).

Another relevant point of comparison is the study by Lin and Xue (2019), which contrasts parser performance for AMR and Elementary Dependency Structures [EDS; Oepen and Lønning (2006)], the original framework from which the bi-lexical DM in our study is derived. Observing a stark differential in parser accuracy, using the same software system, Lin and Xue (2019) seek to identify linguistic phenomena that in the AMR analysis are harder to parse than in EDS. For example, they observe that the more fine-grained AMR classification of different types of named entities (e.g. a token like *Berlin* naming either a location or a person) and the blending of adjectival and modal concepts contribute significantly to increased parsing difficulty for AMR.

3 Data and parsers

3.1 Data and scoring

We focus our first empirical study on the two bi-lexical MRP frameworks (DM and PSD), and the unanchored AMR framework. All presented statistics are against the official evaluation data from the MRP 2019 shared task (Oepen et al., 2019).[1] In the case of DM and PSD, the test set comprises 3359 gold-standard graphs for sentences drawn from the WSJ and Brown corpora; in the case of AMR, 1998 sentences from a variety of sources (newswire, fiction, on-line forums, and Wikipedia). Overall and component-wise MRP evaluation scores, broken down and averaged along different querying dimensions, were computed using an instrumented version of the official scorer, the `mtool` Swiss Army knife of meaning representation.[2]

As our primary dimension of analysis, we consider input complexity in node count, according to the gold-standard target graph for each sentence, over the three selected frameworks. We split the data into decile bins according to node counts, as shown in Fig. 3 (left). We allow for unequal bin sizes, using sentence node count to define cutoff points, so that all sentences with an equal number of nodes are grouped together.

[1] In the case of AMR, this data corresponds to a pre-release of the development section of Abstract Meaning Representation Annotation Release 3.0 (LDC2020T02).

[2] See https://github.com/cfmrp/mtool for details.

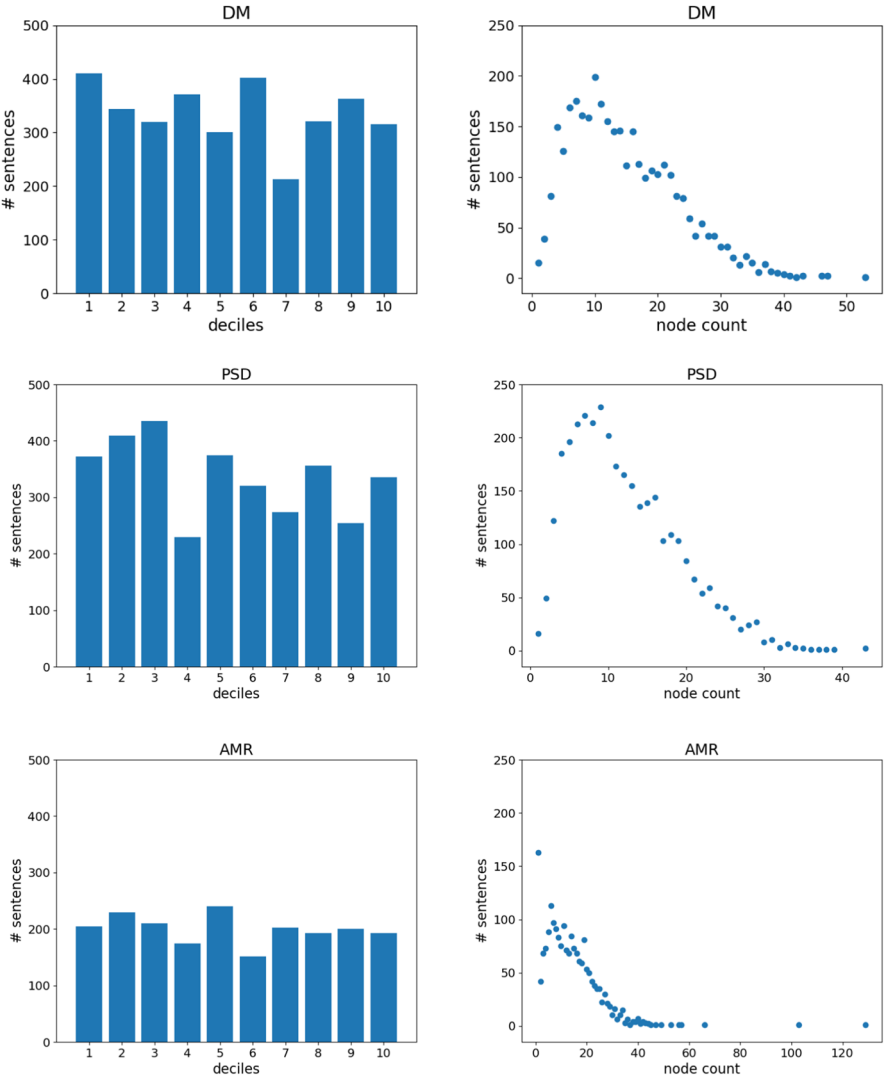

Fig. 3 Left: distribution of sentences by complexity (node count), binned to ten aggregates. Right: absolute counts of sentences by node counts. Top-to-bottom: DM, PSD, AMR

Figure 3 (right) shows the absolute counts of sentences by node count, for each of the three frameworks. While all three frameworks show that the majority of sentences fall between the 5- and 15-node mark, there are two distinct groups of outliers in the case of AMR (Fig. 3, bottom right): one- to two-node sentences, and unusually long sentences (40 nodes and beyond). This leads to the first and last decile bins of AMR data not being comprised of what is conventionally considered a sentence—the first decile bin containing only exceptionally short strings, and the last containing unusually long ones.

Table 1 Sample of outlier sentences by node count in the AMR test data—a very short sentence consisting of metadata, and a very long sentence (actually a list)

1 node	19/01/2010 10:34, 2007-08-27, (End)
129 nodes	Other articles: “A book for Taiwan compatriots-I have a dream”, “The necessity for building memorials museums or temples for Chinese national heroes”, “How long will we tolerate it”, “China is expecting the Olympics, but would never beg for the Olympics", “If we forget the hardship of history, there will be more history of hardship”, “I am Chinese”, “There is only one Chinese nation, there is only one Chinese culture”, “With the over-development of sports, the state may decline”, “’Wealthy country, strong people’ or ’wealthy officials, poor people’?”, “Disgusting Ren Zhiqiang, shut your filthy mouth”, “Hateful housing prices! Pitiable people!”, “Google, you have no right to make irresponsible remarks to China”, “How long will we tolerate the US”, “Japan deserved the bombing-strongly oppose Ban Ki-moon presenting a bouquet in Hiroshima”, “Be strong, my brothers and sisters”, “Do we need low-level entertainment stars, or talents in technological innovation?”, “Contemporary garbage writer, shameless Li Yinhe, please let the children off the hook”

Other examples include mistokenised paragraphs, and sentences with long sequences of geographical names (e.g. weather reports)

A closer look into the data shows that most of these datapoints are noise—metadata, in the case of short sentences; and mistokenised paragraphs and lists, in the case of long ones. A sample is given in Table 1, showing the longest sentence, and one of the shortest. We choose to keep these datapoints, in the interest of remaining in line with the shared task setup, and examining how well different parsers deal with erroneous input. However, when analyzing AMR parser performance in relation to input complexity, we conjecture that these two bins are likely to exhibit unusual behavior.

3.2 Parsing systems

Our choice of models for contrastive evaluation was motivated by the characterisation of systems into three broad families of approaches, as presented, amongst others, by Koller et al. (2019) and Oepen et al. (2019): transition-, factorisation-, and composition-based parsers. Of these, the first two abstractly parallel the two families represented in the study by McDonald and Nivre (2011), whereas composition-based parsing approaches—which compose a semantic structure in a step-wise process guided by a syntactic derivation—are not found in syntactic parsing. We consider participating systems in the MRP 2019 competition, and, within each family of approaches, choose the top-performing systems for the DM, PSD, and AMR frameworks.[3]

Among the transition-based systems in MRP 2019, the best-performing parser is the HIT-SCIR parser (Che et al., 2019), which is also the top-performing parser

[3] We use framework-specific performance on DM, PSD, and AMR—rather than the overall ranking across frameworks within the shared task—as the selection criterion, given that this study is focused on comparing and analysing the results of parsing into these particular frameworks.

Table 2 System scores and rankings in MRP 2019

	MRP score			Ranking			
	P	R	F	Overall	DM	PSD	AMR
HIT-SCIR	.87	.85	.862	1	2	4	2
SJTU-NICT	.87	.83	.853	2	1	3	3
Amazon	.52	.51	.513	8	6	5	1
Saarland	.83	.80	.819	4	4	1	6
Saarland′	.	.	.834	4	4	1	4

The first four columns reproduce the overall MRP shared task results, averaged over all five frameworks, including the three that we have selected for this study

overall; in the factorisation-based family, the SJTU-NICT system (Li et al., 2019) performs best on DM, while the Amazon system (Cao et al., 2019) performs best on AMR; and among the composition-based submissions, the Saarland system (Donatelli et al., 2019) obtains the best PSD results. Table 2 shows the absolute output quality (in terms of MRP precision, recall, and F_1) and the rankings of these systems on the evaluation data, reproducing the official shared task results presented by Oepen et al. (2019).

As reported in Donatelli et al. (2019), there was a post-processing bug in the Saarland system that resulted in invalid labels for named entities on AMR. Saarland′ in Table 2 lists the performance of the corrected version of the system, submitted after the official evaluation period. Since this version demonstrates a significant improvement on the AMR test set (4.3 points), we found it more informative to analyse the AMR output of the fixed system, as opposed to the official submission. For the other two frameworks, the output of Saarland and Saarland′ is essentially the same.

3.2.1 HIT-SCIR

The HIT-SCIR parser is an extended transition-based system designed to predict semantic graphs; it is the overall top-performing system in the shared task. The HIT-SCIR system is built upon the parser of Wang et al. (2018), with a different transition system for each of the featured frameworks. A stacked LSTM architecture is used to model the parsing states, allowing for batch training. The system also incorporates fine-tuned BERT embeddings; additional tagging models for part-of-speech, frame, and lemma decorations; and a pre- and post-processing pipeline to adhere to the MRP format.

3.2.2 SJTU-NICT

The SJTU-NICT parser is a factorisation-based [or "graph-based", in the terminology of McDonald and Nivre (2007)] system, using a feed-forward network and a biaffine attention mechanism for edge and node property predictions on top of BERT embeddings. For the prediction of node-local properties, such as part-of-speech tags and frame labels, the parser also implements a multi-tasking objective. The system

comprises three separate models to handle the three groups of meaning representation included in the shared task—in the case of DM and PSD, an anchoring-based pruning parsing model; and in the case of AMR, a sequence-to-sequence-based parsing model.

3.2.3 Amazon

The Amazon parser is a hybrid system that uses a factorisation-based approach with a latent-alignment mechanism for lexical-anchoring meaning representations (DM, PSD, and AMR), and a CKY parser for phrasal-anchoring representations. By considering AMR as implicitly lexically anchored, an additional alignment search step (Lyu & Titov, 2018) merges AMR with the bi-lexical frameworks for the rest of the pipeline. After alignment, a BiLSTM sequence labelling model with GloVe embeddings assigns concepts to words, identifying nodes. Two BiLSTM encoders are then used to identify heads and dependents, and a biaffine classifier (Dozat & Manning, 2016) predicts labels for the identified edges. Finally, an MSCG inference algorithm (Flanigan et al., 2014) selects the output graph.

3.2.4 Saarland

The Saarland parser, an extension of Lindemann et al. (2019), uses a compositional approach, employing the Apply–Modify Algebra of Groschwitz et al. (2017) to build semantic graphs through highly constrained combinations of smaller graph fragments. A BiLSTM sequence labeling model is used for semantic tagging of word tokens, and the BiLSTM "feature extractor" architecture of Kiperwasser and Goldberg (2016) is employed for predicting dependency trees, with input representations combining ELMo Peters et al. (2018), and BERT Devlin et al. (2019) contextualised word embeddings. Additionally, a decomposition step into subgraphs is necessary for training the model, which is handled using manually defined heuristics.

4 Quantitative study

We perform an empirical study as a first step towards in-depth contrastive analysis of semantic dependency parsing systems. For our initial experiments, we choose to compare the four chosen parsing systems, parsing into the three selected MRP frameworks, and analyzing parser performance depending on input complexity (graph size, in node count).

4.1 Methodology

When considering dimensions in which the accuracy of a meaning representation graph may be analysed, the queries can be separated into two broad categories: (1) structural factors, stemming from formal graph theory—the aspects of a tree or graph such as root node labels, and edge lengths; and (2) linguistic factors—related

to underlying characteristics of the input strings, such as part-of-speech tags, and input complexity.

Drawing from the body of previous work on syntactic dependency tree analysis, we observe a number of dimensions with varying degrees of applicability to semantic dependency graphs. Furthermore, there are two marked differences between syntactic trees and meaning representation graphs that require additional attention. First, from a structural viewpoint, graph nodes allow for multiple incoming edges, as well as outgoing. Secondly, from a linguistically-informed viewpoint, meaning representation graphs also use the concept of node for an additional layer of information, with nodes having particular properties that differ by level of abstraction (in contrast to syntactic dependencies, where nodes are equivalent to tokens, and information on dependency relations is contained in labelled edges).

For a more detailed discussion of potential querying dimensions, we refer to Buljan et al. (2020). In this study, we restrict ourselves to comparing system performace with regards to input complexity. Universally, both syntactic and semantic parsers show lower accuracy for more complex sentences. In the context of MRP, this is true regardless of the level of abstraction of a particular semantic target representation. More complex sentences commonly contain more intricate syntactico-semantic constructions, which call for more parsing decisions to be made, and thus increase the chance of errors, as well as error propagation. In previous work, input complexity has been expressed in terms of length, i.e. the number of tokens. However, word count is less closely related to node count in meaning representation frameworks of higher levels of abstraction, where nodes may represent token substrings (e.g. affixes) or multiple tokens (e.g. multiword expressions), or when there is no clear mapping at all between tokens and graph nodes. An alternative could be to measure sentence input length at character level, but since the semantics of a single word does not necessarily depend on its character count, we rather propose calculating input complexity in terms of nodes in the gold-standard meaning representation graph.

4.2 Experimental results

Figure 4 plots the average P, R, and F_1 scores (i.e. the standard MRP metrics) by input complexity at the node level. We find that the overall results for all three representations—DM, PSD, and AMR—are fairly similar. DM results are in general somewhat higher, but within the same range as PSD, while the unanchored representation, AMR, gives the lowest scores overall. Another notable difference is the decreased performance on the first and last AMR decile bin, due to the nature of the sentences in those bins (semantically irregular fragments and mistokenised paragraphs, as described in Sect. 3.1). We observe no noteworthy variation in the precision vs. recall graphs and will limit subsequent analysis to just the combined F_1 score.

Between all four parsers, we observe only minor differences; overall, behaviour over different input complexities is fairly similar. The Amazon parser

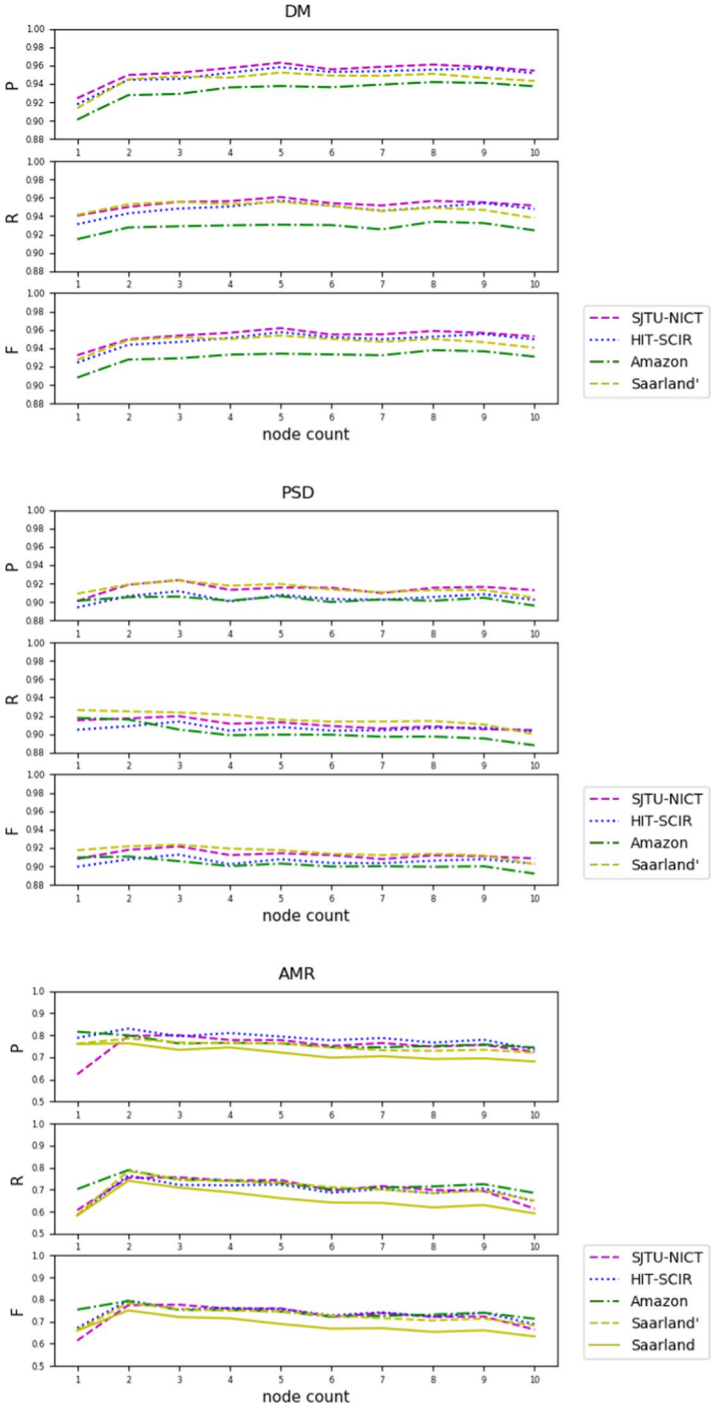

Fig. 4 Overall MRP precision, recall, and F_1 by input complexity for DM, PSD, and AMR

demonstrates lower performance on DM in particular, and less dramatically on PSD, but follows the overall trend over complexity.

At this level of analysis, we do not observe the expected downward trend indicative of a drop in parsing accuracy for more complex sentences. Rather, all three parsers seem relatively robust to input complexity, varying by less than 2 points over input complexity bins. To further study the effects of input complexity on parser performance, we now take advantage of the finer-grained nature of the MRP evaluation metric, breaking down scores according to the different types of information present in meaning representation graphs (see Sect. 3.1 above).

As an indication of graph structure, Fig. 5 plots the (macro-averaged) mean of F_1 scores for top nodes and for (labelled) edges, comparable to labelled attachment score in syntactic dependency parsing evaluation. As we zoom in on just the prediction of graph topology in this perspective, there is a marked drop in accuracy for more complex inputs, across all systems and all three frameworks. This is clear, despite an initial increase in performance which is likely due to particularities of very short sentences (headings, fragments). Here we observe clear differences between the different parsers (Kuhlmann & Oepen, 2016).

While the factorisation-based parser (SJTU-NICT) seems most resilient to the effects of more complex inputs, the degradation is more prominent for the composition-based parser (Saarland), and particularly for the other factorisation-based parser (Amazon), even on the AMR framework for which it demonstrates superior performance. Overall the most successful parser on the PSD framework, the composition-based system (Saarland), nevertheless suffers a drop in performance. Of the three top-performing representatives of each parsing approach, it is the weakest-performing system on bi-lexical frameworks when considering structural information in isolation. Here there is also a clear difference between the two frameworks, where both the Saarland and Amazon parsers exhibit a markedly more dramatic drop in results for DM as compared to PSD. Generally speaking, DM results are on average somewhat lower than the results for PSD, perhaps indicating that DM structural analysis is a harder task. This is possibly related to differences in formal graph properties between these two frameworks.

On the unanchored framework, AMR, the average scores for all four parsers are lower still. There is a marked difference in how the different parsers handle the first decile bin of outlier sentences. Unlike in the case of bi-lexical frameworks, HIT-SCIR shows a dramatic drop in performance in the unanchored framework case—and the top-performing parser on AMR (Amazon) nevertheless struggles the most with structure prediction.

This sample application of the proposed methodology provides directions for future work in the development and cross-framework comparison of semantic parsers; e.g.further development of a particular parser may wish to focus on improving the system's structural capacities, or for a different parser its accuracy regarding different types of node decorations. More extensive testing is also needed to evaluate the role of training differences, i.e. the use of contextualised word embeddings (common across all three parsers) versus decomposition heuristics (specific to the composition-based system) in fluctuation across input complexity.

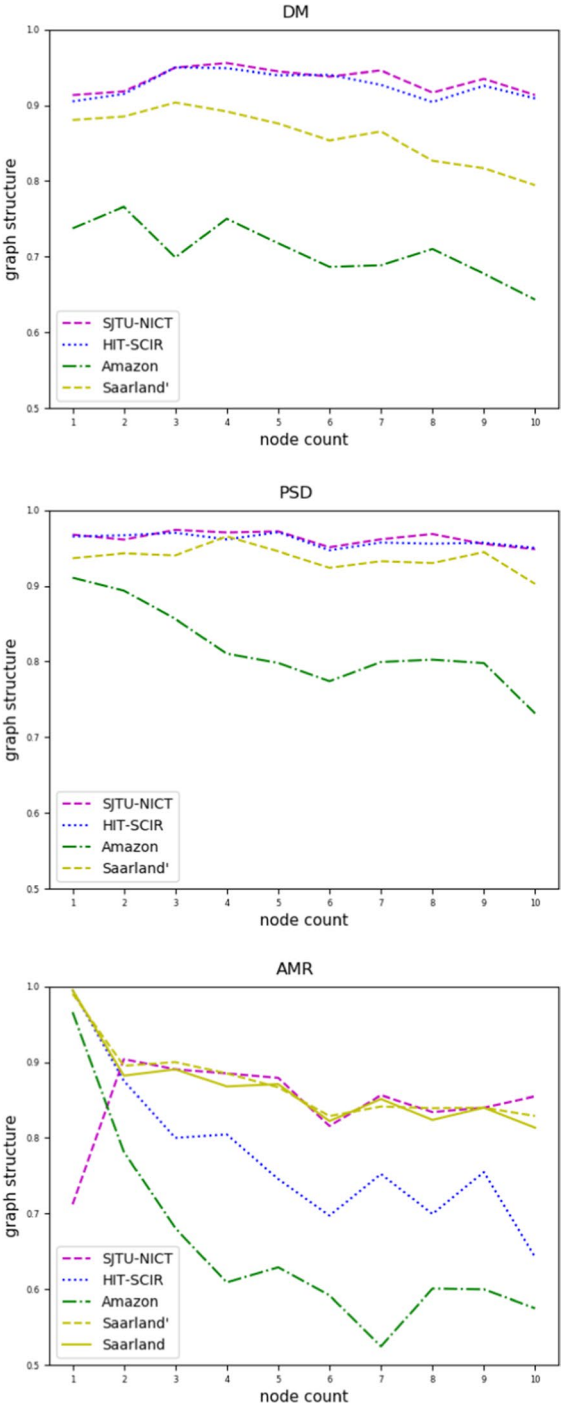

Fig. 5 Structural F_1 (edges and top nodes) by input complexity for DM, PSD, and AMR

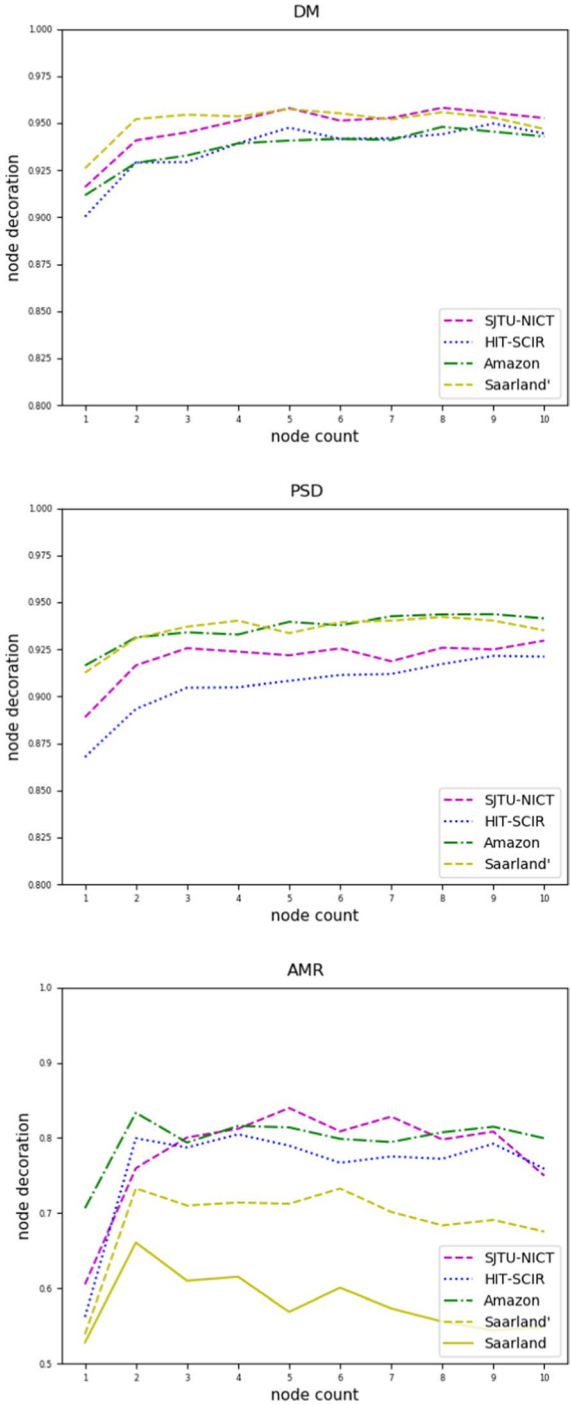

Fig. 6 Node-local F_1 (labels and properties) by input complexity for DM, PSD, and AMR

So far, these findings are largely in line with those of previous studies, most notably Kulmizev et al. (2019)—demonstrating that this dimension of analysis is indeed applicable to semantic dependency parsing. These preliminary observations could also point to a similar trend as seen with the introduction of neural networks to syntactic parsing: narrowing of the margin of difference in model performances.

By contrast, Fig. 6 plots system performance considering only the mean of F_1 scores for node-local information, labels and node properties—a measure of how accurately the parsers decorate the graph nodes. As discussed in Sect. 2, this concept has no clear equivalent in syntactic dependency parsing. Here we observe a similar trend for all three parsers across the two bi-lexical representations; the prediction of node-local information does not seem to be notably affected by input complexity and is fairly stable over sentences of increasing complexity. It is also clear that the Saarland parser, which is the top-performing system for the PSD representation, outperforms the other parsers for the task of node decoration, compensating in large parts for its weaker performance in graph structure prediction.

This is reversed in the case of the unanchored AMR representation, where the Saarland parser (both the submitted system version, and the version with corrected NER) demonstrates a much worse performance overall with regards to node-local information. Again, for all parsers, the trends depending on input complexity are similar, including the handling of outliers in the first AMR decile bin. Both factorisation-based parsers (SJTU-NICT and Amazon), while showing different trends and performance on all other frameworks and scoring metrics, are comparatively matched on the AMR framework, considering node-local information.

The quantitative study presented here is an adaptation of the methodology previously applied to syntactic dependency parsing systems, applied to semantic dependency parsers. More insights were gleaned by breaking down the analysis along structural vs. node-local information of the meaning representation graphs, and motivated further in-depth analysis, as an attempt to categorise error types and identify common error patterns across systems. We present this next step in the following section.

5 Qualitative analysis

In the second part of our study, we transition from quantitative batch analysis of error trends towards a more sample-focused view—combining a top-down and bottom-up approach to quantifying error patterns and identifying linguistic phenomena that are particularly challenging to parse, for all systems or in individual cases.

In Sect. 5.1, we propose a methodology for identification of sentences that prove to be difficult to parse. In Sect. 5.2, we manually inspect the parser outputs for these problematic sentences, and analyse some of the discrepancies between the gold standard and select system outputs. Building on this per-sample inspection, we begin to generalise and identify error trends across frameworks and systems, and give a summary of our findings in Sect. 5.3.

Table 3 Percentage and count of *troublemaker* sentences—sentences falling below 2 std.dev. of decile-mean F_1; breakdown per parser and decile bin

framework	DM				PSD				AMR			
parser / decile	SJTU	Saar	Amaz	HIT	SJTU	Saar	Amaz	HIT	SJTU	Saar	Amaz	HIT
1	5.36	5.12	5.36	5.14	3.76	5.10	4.83	5.91	0.00	0.00	3.41	0.00
2	5.52	5.81	6.39	5.52	4.15	4.15	4.88	4.64	6.11	3.93	3.05	4.36
3	4.68	5.00	6.48	5.31	4.59	5.76	4.59	6.20	3.33	4.76	3.33	2.85
4	4.58	5.12	3.50	4.85	3.49	4.36	4.80	3.49	1.72	2.87	3.44	4.02
5	4.66	5.00	4.00	4.33	3.46	2.66	5.06	3.46	4.16	2.08	4.16	3.75
6	4.72	3.73	3.73	5.72	3.75	4.06	2.81	4.37	1.97	1.97	1.97	3.94
7	6.60	3.77	3.77	5.18	4.01	4.74	4.37	4.74	2.47	3.46	2.97	4.45
8	3.73	4.04	5.29	4.67	4.77	4.21	3.37	3.65	4.14	2.07	2.07	3.10
9	4.13	4.40	4.13	4.68	3.14	3.14	2.75	4.33	2.50	2.50	2.50	4.00
10	3.48	2.22	3.79	4.74	2.98	2.69	2.38	3.58	4.66	4.14	2.59	3.62
all deciles	4.70	4.47	4.50	5.03	3.87	4.14	4.05	4.53	3.20	2.80	3.00	3.40
$\sum$ sentences	*158*	*150*	*151*	*169*	*130*	*139*	*136*	*152*	*64*	*56*	*60*	*68*

Deciles with higher troublemaker percentages highlighted in darkening blue

5.1 Troublemaker sentences

Considering all parser-generated meaning representation graphs that do not align perfectly with the gold graphs, the first challenge in identifying error patterns is choosing where to focus the investigative effort, i.e. how to define the subset of data-points to manually investigate. As a first step, we select sentences that are shown to be the most difficult to parse for all four parsers. We define these as sentences for which, across all parsers, and within a framework, the F_1 of every parser's output falls below 2 standard deviations of that parser's mean F_1 for the decile bin to which the sentence belongs. Table 3 shows the percentages and counts of these sentences, broken down by parsers and decile bins.

Intuitively, there is merit in taking a closer look at those sentences that are shown to be challenging for all systems, as this may reveal more about certain framework features or linguistic phenomena that are universally difficult for parsers to replicate or interpret. For simplicity, we further refer to these sentences as "troublemakers".

5.1.1 Parser-specific outliers

As discussed in previous sections, and demonstrated by the experimental results in Sect. 4, we can assume that there is a rough ranking to how challenging a particular framework is to parse into—DM being the least, and AMR most difficult of the three. This is also partially based on inter-annotator agreement reports (Banarescu et al., 2013; Bender et al., 2015) which show that fully manual annotation (such as that of AMR) leads to higher degrees of freedom, and thus less inter-annotator agreement, than annotations done on existing automatically parsed layers (such as for DM and PSD). These annotation discrepancies can be expected to result in a higher proportion of errors made by AMR parsers, as opposed to DM or PSD.

However, there is a discrepancy between this expectation, and the actual proportion of sentences that are identified as troublemakers for the different parsers.

Table 4 Top-performing parser F_1 vs. percentage of troublemaker sentences in total test set, per framework

Framework	Best F	% of trouble-makers
DM	95.5	1.22
PSD	91.8	0.77
AMR	73.4	0.40

Table 5 Number of parser-specific outliers, by deciles (DM/PSD/AMR)

	DM				PSD				AMR			
	SJTU	Saar	Amaz	HIT	SJTU	Saar	Amaz	HIT	SJTU	Saar	Amaz	HIT
1	14	4	20	6	4	8	8	16	0	0	14	0
2	6	6	12	8	12	10	12	16	18	8	6	14
3	4	6	14	8	4	16	8	16	6	12	6	4
4	6	12	10	10	2	8	10	2	0	2	4	8
5	8	10	6	8	10	6	18	6	0	2	4	8
6	16	14	16	30	10	10	2	10	14	2	14	12
7	8	2	2	4	8	12	4	4	0	4	2	8
8	2	6	10	10	8	4	6	8	8	4	4	8
9	6	14	12	8	6	4	4	12	6	8	6	10
10	8	6	10	8	10	12	8	14	10	8	8	8
$\sum$	78	80	112	100	74	90	76	104	62	50	64	80
%	2.3	2.3	3.3	2.9	2.2	2.6	2.2	3.0	5.1	4.1	5.3	6.9

Looking at Table 3, the percentage of troublemakers in the dataset hovers around the 4–5% mark on DM and PSD, and 3% on AMR—which seems to indicate that the different systems produced more highly erroneous graphs on the "easier" frameworks. This is further highlighted by Table 4, which lists the F_1 scores of the highest-performing parsers per framework, against the proportion of joint troublemaker sentences for all parsers in the dataset. Counter-intuitively, the framework on which the best-performing parser achieves the highest F_1 of all systems is also the framework with the highest proportion of sentences for which all parsers gave a highly erroneous output.

This inverse relationship implies that there is more homogeneity in parser errors on DM and PSD, and more diversity in errors produced when parsing into AMR. In other words, there seem to be more sentences that all systems parse exceptionally badly on DM and PSD, producing incorrect output for the same input. Meanwhile, on AMR the different systems struggle to produce outputs for different types of sentences, hence the smaller number of troublemakers (sentences on which all four parsers fail drastically).

In order to further investigate these parser-specific errors in more detail, we introduce the notion of "outliers"—lowest-scoring sentences that are particularly

bad for one parser, but not the others. Where global troublemakers may reveal error patterns across parsers and frameworks, these parser-specific outliers may help pinpoint where parsers diverge in the likelihood of producing serious errors.

Table 5 lists the counts of these parser-specific outlier sentences. It is visible that AMR has a significantly higher proportion of these difficult sentences in the test set than either DM or PSD, explaining the previously seen discrepancy between system performance and the proportion of troublemakers in the evaluation set.

5.1.2 Error fingerprinting

The aim of the qualitative step of our error analysis is also to manually inspect error-prone samples and identify sources of uncertainty for particular frameworks or parsers, and thus complement the conclusions drawn from the full-scale quantitative analysis. While the quantitative perspective given by Tables 3 and 5 sheds light on error frequencies for parsers and frameworks, it is still unclear whether, and what makes, these parsers fail on certain samples in the same way. This is the sort of question that manual inspection might provide more hypotheses for. However, it is visible from Table 3 that even when narrowing down the pool of misparsed sentences through the definition of *troublemakers*, the number of these samples would make manual inspection overly time-consuming. In order to further focus our view on a handful of samples that would be informative enough to warrant manual inspection, we introduce the concept of "error fingerprinting".

During scoring, a parser's output is defined by three properties: node labels, edges, and anchors. These are the properties according to which a system-produced graph is compared to the gold graph. Subsequently, an `mtool` error trace for a system-produced graph is a list of those labels, edges, and anchors that are either missing or surplus in the system graph, compared to the gold graph. (A more detailed explanation of error tracing is given in the following subsection, Sect. 5.2). This list of errors—more detailed than an overall score of a particular system graph—enables identifying common error patterns between parsers, and so, possibly, common challenges in frameworks or sentences.

We use these points of disagreement to define an "error fingerprint" of a sentence parse. As the scorer produces an error trace listing missing or surplus elements of a sentence's system-produced graph, we collect these errors into a tuple that makes up the error fingerprint.

For example, given the sentence "You know – a muzzle for my sheep...", the Amazon system's AMR graph would generate the following error fingerprint:

```
(lm_1 muzzle-01) (lm_0 thing) (lm_3 i) (ls_know-01)
(ls_you) (ls_muzzle)
 (em_1 0 arg2) (em_1 2 arg1) (es_arg0) (es_beneficiary),
```

with three missing and three surplus labels, and two missing and two surplus edges. This example is further discussed below and illustrated in Fig. 7.

For each sentence, we measure the percentage of overlap between error fingerprints, pairwise across the systems. We define a threshold of 90% overlap between

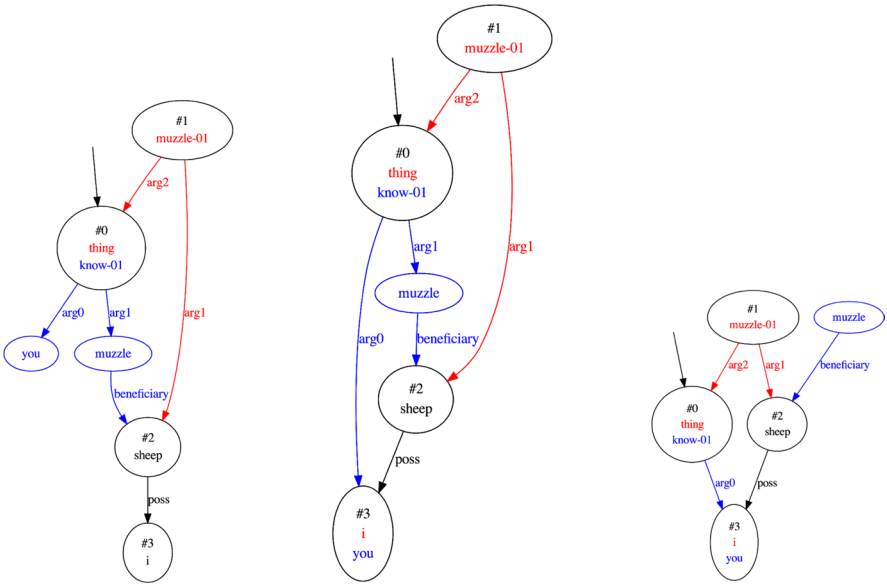

Fig. 7 "You know—a muzzle for my sheep..."—AMR graphs as constructed by the four parsers (left to right: HIT-SCIR and Saarland (same output), SJTU-NICT, Amazon)

error fingerprints as the point at which similarities between the errors produced by two systems are high enough to warrant manual inspection. All sentences with 90% or greater overlap, for any pair of parsers, are considered particularly challenging, and we further analyse parser outputs for these sentences in the remainder of this section.

5.2 Gold vs. parser graphs

Using `mtool`, we visualise the differences between gold parse graphs and system outputs for troublemaker sentences with substantially high error fingerprint overlaps. In these visualisations, false negatives are drawn in red—e.g.nodes existing in the gold graph, but missing from the system-produced graph. False positives are drawn in blue—e.g.surplus nodes introduced by the system, but nonexistent in the gold graph. Outlines and labels drawn in black show where the system and gold graphs are in agreement. The following figures present a handful of examples for this step of the qualitative analysis.

Figure 7 shows the AMR parsing graphs produced by the systems for the sentence "You know – a muzzle for my sheep ...". (Note that HIT-SCIR and Saarland produced identical outputs for this example (leftmost graph), so only three unique graphs are shown here.) A common error shared across all four parsers is a literal interpretation of *you know*—a semantically vacuous discourse connective in the gold standard. This is visible from the blue *know-01* label present in all three graphs: the blue colour marks the node label as surplus, the label itself denotes

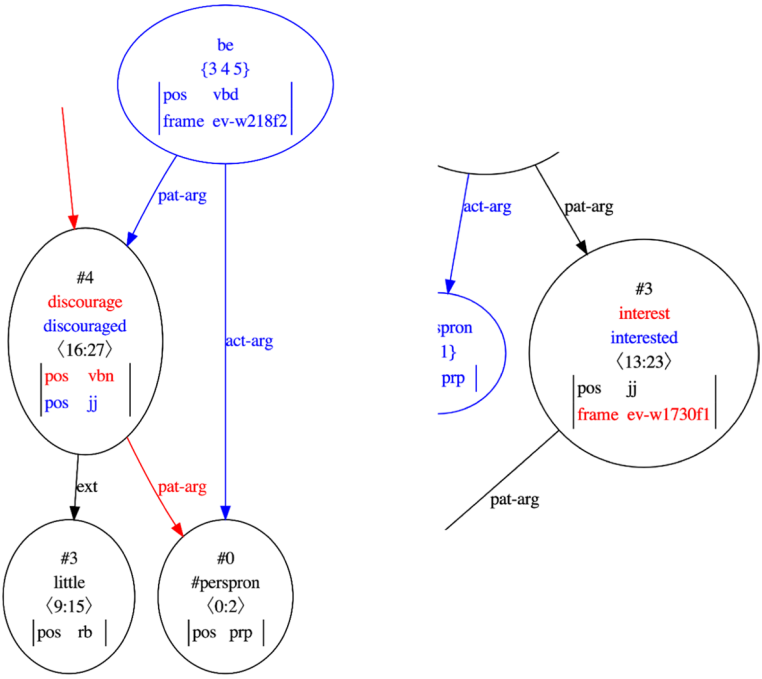

Fig. 8 "He was a little discouraged." (all systems); "It's getting interested in something that counts." (all except SJTU-NICT; cropped selection)—deverbal adjectives in PSD

the frame identifier for the verb "know", and the position of the label—assigned to the top node—marks "know" as the main predicate.

Furthermore, likely as a result of falsely identifying *know* as the main predicate, all four systems fail to correctly produce the deverbalised nominalisation of *muzzle* as annotated in the AMR gold graph. The red *muzzle-01* label, existing in the gold graph, is the frame of the verb "muzzle". A red edge connects it to *sheep*, its patient argument (arg1, the thing that is muzzled); another red edge connects it to the abstract *thing* (arg2, denoting the *thing* as the entity that is doing the muzzling). Instead, all four systems introduce a blue node for *muzzle* as the object (arg1) of "know": "You know a muzzle".

Figure 8 is an example of a common error found in the outputs for PSD—but not in the other frameworks—missed verbalisation in the case of verbal adjectives. This is caused by the ambiguity for adjectival participles between a verbal use (as in the example sentence "He was a little *discouraged*" in Fig. 8) and an adjectival use, as illustrated by *interested* in the left graph fragment in Fig. 8.

Another common challenge, shown on the example of DM in Fig. 9, but frequent across all parsers and frameworks, is a literal interpretation of the existential *there* as a location marker. In this example, both parsers suggest a locative reading, visible from the blue (surplus) *loc* relation between *there* and *dignity* (bottommost node in both graphs).

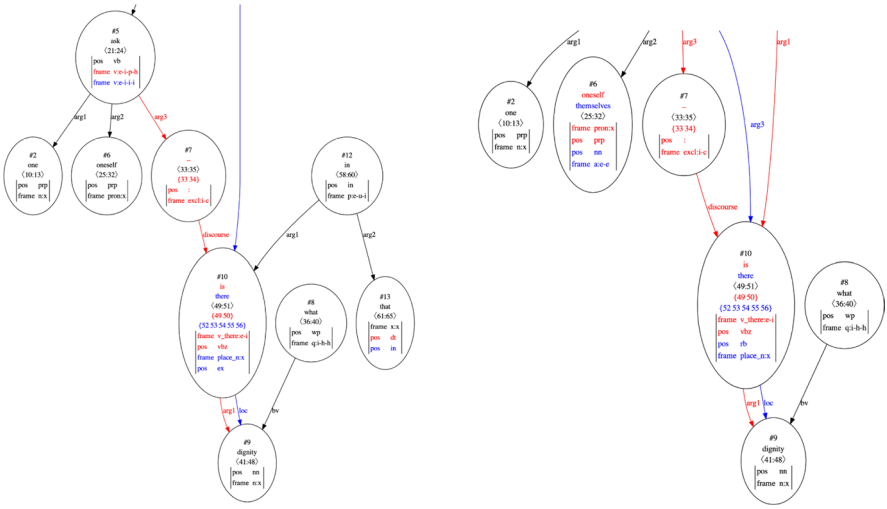

Fig. 9 "Really – one has to ask oneself – what dignity is there in that?" (Saarland (left) and HIT-SCIR (right))—existential "there" vs. location

We apply this analytical technique to all troublemaker sentences identified in Sect. 5.1, and summarise our findings in the following section.

5.3 Error analysis

Finally, we provide a qualitative summary of the error fingerprinting method for analysis. Our aim is to propose how to distill common errors and general error patterns among the parsers. We organise this analysis by framework, across parsing systems.

5.3.1 DM

As highlighted in Fig. 7 on the AMR example, all four parsers output literal interpretations of discourse fragments for the DM framework as well. Furthermore, we find that all parsers struggle with multiword expressions (and multiword named entities) when parsing into DM (e.g. "I shall look *as if* I were suffering."; "Analysts at *Standard & Poor's* say junk bond offerings by 'tightly stretched' issuers seem to be growing.") All four parsers also incorrectly position conjunction relations between nodes, treat auxiliary verbs as the top predicate, and interpret the existential *there* as a location (as in Fig. 9, or "It seemed to me, even, that *there* was nothing more fragile on all Earth."). The SJTU-NICT and Saarland systems show similar error patterns, where their output stands out from the other systems (e.g. misidentifying arguments across subclauses). HIT-SCIR, in particular, introduces surplus relations to avoid disconnected node clusters, and "fixes" out-of-vocabulary words by matching them to the character-wise nearest neighbour ("It sounds like with the [*Rubens* → *rubenstein*] he got absolutely taken to the cleaners.") Interestingly, in cases where

the gold annotation is questionable (incorrect, or difficult to justify without context), parsers are in agreement with their presumably erroneous output.

5.3.2 PSD

Similarly to the case of DM, all parsers give literal interpretations of discourse fragments, locative interpretations of adverbs such as *where* and the existential *there*, and non-modal analyses of modal verbs, etc. ("That's *where* device quackery *can* lead."). Additionally, all parsers frequently treat auxiliary verbs as top predicates, and invert actor and patient roles. Saarland and HIT-SCIR inconsistently parse deverbal adjectives, as shown in Fig. 8, and introduce spurious MWEs ("Because I *am about* to die of thirst..."). Again, HIT-SCIR finds replacements for out-of-vocabulary words ("Maybe that's your [*forte* → *rationale*].")

5.3.3 AMR

When parsing into AMR, too, all parsers output literal interpretations of discourse fragments. Furthermore, all parsers struggle with conditionals—e.g. misidentifying conditional subclauses ("Work hard if you really want out.") or failing to identify causal relations ("You can't give in to that or it will just escalate."). SJTU-NICT and the Amazon parser, in particular, have a tendency to conflate arguments denoted by pronouns, as in the example in Fig. 7 (*i*/*you*).

The qualitative analysis presented in this section is a sample of the kind of in-depth error analysis we outline with the methodology proposed above. We move from high-level quantitative analysis of parser performance to a statistical overview of error distribution. After identifying portions of the dataset that are most challenging to parse for all systems, we also identify parser-specific outliers. We conclude that there is more diversity in errors produced when parsing into AMR, and further confirm this with manual analysis of most frequent error patterns. While some errors are a result of of linguistic phenomena for which an incorrect graph is more likely to be construed, others may be a result of inconsistent annotation within frameworks.

Along with the previous finding that the system outputs are more prone to errors going from DM, over PSD, to AMR, these annotation-related findings are tentatively supported by the small inter-annotator agreement study by Bender et al. (2015), which shows that grammar-based semantic annotation increases IAA compared to fully manual annotations. In the case of the three frameworks used in our study, DM and PSD (with annotation over an automatically generated grammar/syntactic layer) stand apart from AMR (with fully manual annotation, resulting in higher degrees of freedom), which partially explains the diversity of errors seen in automatic AMR parses, and less overlap in troublemaker sentences between the two sets of frameworks.

In order to further explore the possible origins of some other common errors, e.g. error patterns dependent on system architecture or parsing approach, an ablative study of target system components is needed. We leave this to future work, as an invitation to parser developers interested in applying these methods to their own systems. On the other hand, error patterns based on annotation discrepancies are also

informative for developers of semantic frameworks and their respective annotation guidelines.

6 Conclusion

The main motivation behind this work has been the development of a methodology for diagnostic evaluation and error analysis of meaning representation parsers that moves beyond the aggregated metrics commonly reported for this task. In order to gain an understanding of the types of errors found in state-of-the-art parsers, we have presented a quantitative and qualitative error analysis that contrasts four different parsers across three meaning representation frameworks.

In this study, we have taken as our point of departure the datasets and top-performing systems in the MRP shared task of 2019. In our analysis, we included both the bi-lexical frameworks of DM and PSD (which bear some formal, if not linguistic, similarity with common syntactic dependency graphs), as well as the AMR framework, which due to its lack of lexical anchoring presents a more "free-floating" approach to meaning representation. Across these three frameworks, we have further selected representatives of the three main modelling approaches to the task of MRP: transition-based (represented by the HIT-SCIR parser), factorisation-based (the SJTU-NICT parser), and composition-based (the Saarland parser) architectures, as well as a parser originally developed for AMR parsing (the Amazon parser), which also adopts a factorisation-based approach.

The quantitative study presented in Sect. 4 adapted the type of analysis previously applied to the diagnostic study of syntactic dependency parsers to the task of MRP.

Our analysis initially showed that comparisons of the standard measures of accuracy for this task (precision, recall, and F_1) across different input complexities (as indicated by gold-standard node count) reveals only minor differences between the parsers across frameworks.

We therefore proposed to break down the semantic graphs and rather analyse the accuracy of the parsers along structural vs. node-local dimensions of the three frameworks. Our analysis of top node and edge F_1, capturing structural properties of the graphs, revealed a clear drop in accuracy for more complex inputs. The analysis further allowed us to observe clear differences between frameworks and parsers. We observed that the results for the prediction of structural properties are generally lower for DM than PSD, providing an indication that this is a structurally somewhat more difficult target representation.

Somewhat surprisingly perhaps, we also found that the Saarland parser, which was the top performing parser for the PSD framework, showed a relatively weak performance for the bi-lexical frameworks in terms of graph structure prediction, coupled with much stronger node decoration performance. For AMR, we also found that the Amazon parser does not excel at the prediction of structure and is in fact the weakest-performing parser.

We then went on to analyse the prediction of node-local information, where we found that for these types of properties, the parsers are not generally affected by input complexity. Here we also observed differences between the parsers across

different frameworks, albeit somewhat less clear. We found that the Saarland parser largely outperforms the others on PSD, but clearly struggles on the AMR data set, where the factorisation-based parsers (Amazon and SJTU-NICT) yield the strongest results.

The more detailed breakdown of different graph properties provided several insights into differences between the parsers and meaning representation frameworks, but also revealed the need for a more detailed error analysis of system outputs across frameworks. In Sect. 5 we therefore proposed a methodology for the identification of data points for further manual inspection.

We defined what we call "troublemaker" sentences to be inputs that proved difficult for all the parsers, and parser-specific "outliers" to be sentences that were error-specific to only one parser. At this level of analysis, we observed a clear difference between the frameworks showing that the AMR results contain a larger proportion of parser-specific errors.

We further proposed a notion of "error fingerprints" of troublemakers to identify individual sentences in each framework that proved difficult to parse. In Sect. 5.3 we illustrated the use of this methodology in the qualitative error analysis of semantic graphs. By selecting sentences using our proposed methodology and visualisation of differences between gold and system outputs, we manually inspected all troublemaker sentences and provided a summary of common error types for the different parsers across frameworks. We observed that some error types recur across frameworks, such as the treatment of discourse phenomena and non-referential entities, whilst others are specific to a set of parsers or a particular parser, e.g. the observed HIT-SCIR substitutions for out-of-vocabulary words.

We conclude that the methodology proposed above has allowed for useful insights into the frameworks and parsers represented in the 2019 MRP shared task. We believe that comparisons of error fingerprints for troublemaker sentences provide an efficient methodology for analysis of system errors that cuts across different frameworks and modelling strategies. Even so, there are limitations to the conclusions that the methodology illustrated here allows for, and these open additional avenues for future research. In particular, analysis that aims to connect specific system architectures or modeling strategies with error profiles remains unexplored. This would require more experimental studies that isolate certain system components systematically and seek to explicitly relate performance differences with observed system errors. The methodology presented here provides an important tool in the future development of improved semantic dependency parsers.

Funding Open access funding provided by University of Oslo (incl Oslo University Hospital).

References

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., & Schneider, N. (2013). Abstract Meaning Representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, Sofia (pp. 178–186). http://www.aclweb.org/anthology/W13-2322

Bender, E. M., Flickinger, D., Oepen, S., Packard, W., & Copestake, A. (2015). Layers of interpretation: On grammar and compositionality. In *Proceedings of the 11th international conference on computational semantics* (pp. 239–249).

Buchholz, S., & Marsi, E. (2006). CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the 10th conference on natural language learning*, New York, NY (pp. 149–164). http://www.aclweb.org/anthology/W/W06/W06-2920

Buljan, M., Nivre, J., Oepen, S., & Øvrelid, L. (2020). A tale of three parsers: Towards diagnostic evaluation for meaning representation parsing. In *Proceedings of the 12th language resources and evaluation conference* (pp. 1902–1909).

Cai, S., & Knight, K. (2013). Smatch: An evaluation metric for semantic feature structures. In *Proceedings of the 51th meeting of the Association for Computational Linguistics*, Sofia (pp. 748–752). http://www.aclweb.org/anthology/P13-2131

Cao, J., Zhang, Y., Youssef, A., & Srikumar, V. (2019). Amazon at MRP 2019: Parsing meaning representations with lexical and phrasal anchoring. In *Proceedings of the shared task on cross-framework meaning representation parsing at the 2019 conference on computational natural language learning*, Hong Kong (pp. 138–148).

Che, W., Dou, L., Xu, Y., Wang, Y., Liu, Y., & Liu, T. (2019). HIT-SCIR at MRP 2019: A unified pipeline for meaning representation parsing via efficient training and effective encoding. In *Proceedings of the shared task on cross-framework meaning representation parsing at the 2019 conference on computational natural language learning, Hong Kong* (pp. 76–85).

Damonte, M., Cohen, S. B., & Satta, G. (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, Association for Computational linguistics, Valencia* (pp. 536–546). https://aclanthology.org/E17-1051

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, MN

Donatelli, L., Fowlie, M., Groschwitz, J., Koller, A., Lindemann, M., Mina, M., & Weißenhorn, P. (2019). Saarland at MRP 2019: Compositional parsing across all graphbanks. In *Proceedings of the shared task on cross-framework meaning representation parsing at the 2019 conference on computational natural language learning*, Hong Kong (pp. 66–75).

Dozat, T., & Manning, C. D. (2016). Deep biaffine attention for neural dependency parsing. arXiv preprint http://arxiv.org/abs/161101734

Dridan, R., & Oepen, S. (2011). Parser evaluation using elementary dependency matching. In *Proceedings of the 12th international conference on parsing technologies*, Dublin (pp. 225–230).

Flanigan, J., Thomson, S., Carbonell, J., Dyer, C., & Smith, N. A. (2014). A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd meeting of the Association for Computational Linguistics*, Baltimore, MD (pp. 1426–1436). http://www.aclweb.org/anthology/P14-1134

Groschwitz, J., Fowlie, M., Johnson, M., & Koller, A. (2017). A constrained graph algebra for semantic parsing with AMRs. In *Proceedings of the 12th international conference on computational semantics*, Montpellier.

Hajič, J., Hajičová, E., Panevová, J., Sgall, P., Bojar, O., Cinková, S., Fučíková, E., Mikulová, M., Pajas, P., Popelka, J., Semecký, J., Šindlerová, J., Štěpánek, J., Toman, J., Urešová, Z., & Žabokrtský, Z. (2012). Announcing Prague Czech-English Dependency Treebank 2.0. In *Proceedings of the 8th*

international conference on language resources and evaluation, Istanbul (pp. 3153–3160). http://www.lrec-conf.org/proceedings/lrec2012/pdf/510_Paper.pdf

Ivanova, A., Oepen, S., Øvrelid, L., & Flickinger, D. (2012). Who did what to whom? A contrastive study of syntacto-semantic dependencies. In *Proceedings of the 6th linguistic annotation workshop*, Jeju (pp. 2–11).

Kate, R. J., & Wong, Y. W. (2010). Semantic parsing. The task, the state of the art and the future. In *Tutorial abstracts of the 20th meeting of the Association for Computational Linguistics*, Uppsala (p. 6). http://www.aclweb.org/anthology/P10-5006

Kiperwasser, E., & Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics, 4*, 313–327.

Koller, A., Oepen, S., & Sun, W. (2019). Graph-based meaning representations. Design and processing. In *Proceedings of the 57th meeting of the Association for Computational Linguistics: Tutorial Abstracts*, Florence (pp. 6–11). https://doi.org/10.18653/v1/P19-4002

Kuhlmann, M., & Oepen, S. (2016). Towards a catalogue of linguistic graph banks. *Computational Linguistics, 42*(4), 819–827.

Kulmizev, A., de Lhoneux, M., Gontrum, J., Fano, E., & Nivre, J. (2019). Deep contextualized word embeddings in transition-based and graph-based dependency parsing: A tale of two parsers revisited. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing*, Hong Kong (pp. 2755–2768).

Li, Z., Zhao, H., Zhang, Z., Wang, R., Utiyama, M., & Sumita, E. (2019). SJTU–NICT at MRP 2019: Multi-task learning for end-to-end uniform semantic graph parsing. In *Proceedings of the shared task on cross-framework meaning representation parsing at the 2019 conference on computational natural language learning*, Hong Kong (pp 45–54).

Lin, Z., & Xue, N. (2019). Parsing meaning representations: Is easier always better? In *Proceedings of the first international workshop on designing meaning representations, Association for Computational Linguistics*, Florence (pp. 34–43). https://doi.org/10.18653/v1/W19-3304, https://www.aclweb.org/anthology/W19-3304

Lindemann, M., Groschwitz, J., & Koller, A. (2019). Compositional semantic parsing across graphbanks. In *Proceedings of the 57th annual meeting of the Association for Computational Linguistics*, Florence (pp. 4576–4585).

Lyu, C., & Titov, I. (2018). AMR parsing as graph prediction with latent alignment. arXiv preprint retrieved from http://arxiv.org/abs/180505286

McDonald, R., & Nivre, J. (2007). Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and conference on natural language learning*, Prague.

McDonald, R., & Nivre, J. (2011). Analyzing and integrating dependency parsers. *Computational Linguistics, 37*(1), 197–230.

Miyao, Y., Oepen, S., & Zeman, D. (2014). In-house: An ensemble of pre-existing off-the-shelf parsers. In *Proceedings of the 8th international workshop on semantic evaluation*, Dublin (pp. 63–72).

Oepen, S., Abend, O., Abzianidze, L., Bos, J., Hajič, J., Hershcovich, D., Li, B., O'Gorman, T., Xue, N., & Zeman, D. (2020). MRP 2020: The second shared task on cross-framework and cross-lingual meaning representation parsing. In *Proceedings of the CoNLL 2020 shared task: Cross-framework and cross-lingual meaning representation parsing, online* (pp. 1–22).

Oepen, S., Abend, O., Hajič, J., Hershcovich, D., Kuhlmann, M., O'Gorman, T., Xue, N., Chun, J., Straka, M., & Urešová, Z. (2019). MRP 2019: Cross-framework Meaning Representation Parsing. In *Proceedings of the shared task on cross-framework meaning representation parsing at the 2019 conference on computational natural language learning*, Hong Kong (pp. 1–27).

Oepen, S., Kuhlmann, M., Miyao, Y., Zeman, D., Cinková, S., Flickinger, D., Hajič, J., & Urešová, Z. (2015). SemEval 2015 Task 18. Broad-coverage semantic dependency parsing. In *Proceedings of the 9th international workshop on semantic evaluation*, Denver, CO (pp. 915–926).

Oepen, S., Kuhlmann, M., Miyao, Y., Zeman, D., Flickinger, D., Hajič, J., Ivanova, A., & Zhang, Y. (2014). SemEval 2014 Task 8. Broad-coverage semantic dependency parsing. In *Proceedings of the 8th international workshop on semantic evaluation*, Dublin (pp. 63–72). http://www.aclweb.org/anthology/S15-2153

Oepen, S., & Lønning, J. T. (2006). Discriminant-based MRS banking. In *Proceedings of the 5th international conference on language resources and evaluation*, Genoa (pp. 1250–1255).

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (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)*, New Orleans, LA (pp. 2227–2237).

Wang, Y., Che, W., Guo, J., & Liu, T. (2018). A neural transition-based approach for semantic dependency graph parsing. In *Thirty-second AAAI conference on artificial intelligence*.