Using MRS for Semantic Representation in Task-Oriented Dialogue

Denson George and Baber Khalid and Matthew Stone
Rutgers University
Piscataway, NJ 08854
firstname.lastname@rutgers.edu

Abstract

Task-oriented dialogue (TOD) requires capabilities such as lookahead planning, reasoning, and belief state tracking, which continue to present challenges for end-to-end methods based on large language models (LLMs). As a possible method of addressing these concerns, we are exploring the integration of structured semantic representations with planning inferences. As a first step in this project, we describe an algorithm for generating Minimal Recursion Semantics (MRS) from dependency parses, obtained from a machine learning (ML) syntactic parser, and validate its performance on a challenging cooking domain. Specifically, we compare predicate-argument relations recovered by our approach with predicate-argument relations annotated using Abstract Meaning Representation (AMR). Our system is consistent with the gold standard in 94.1% of relations.

1 Introduction

Natural Language Understanding (NLU) is a core capability of all dialogue systems. It enables machines to interpret and generate contextually appropriate responses to language. Semantic parsing has long been a crucial component of NLU, providing an early-stage component for converting language into a structured semantic representation. However, since the emergence of large language models (LLMs), there has been a trend towards entirely replacing NLU modules and structured semantic representations with end-to-end model inference (OpenAI, 2022). Such systems have been shown to perform well in question answering, natural language generation (NLG), translation, summarization, and many other applications (OpenAI, 2024). Nevertheless, state-of-the-art Task-Oriented Dialogue (TOD) systems still benefit from an NLU module or a semantic representation (Feng et al., 2021; Zhu et al., 2023; Sun et al., 2023), and outperform single-call LLM systems in specific TOD benchmarks (Hudeček and Dusek, 2023). LLMs struggle with key aspects of TOD, including lookahead planning problems (Bachmann and Nagarajan, 2024), reasoning (Jiang et al., 2024), and tracking belief states (Chiu et al., 2023). These issues highlight the potential advantages of having a structured semantic representation that can be updated based on dialogue, information from the environment, and plan-based task reasoning (Geib et al., 2022).

In this paper, we explore MRS as a semantic representation framework due to its rich expressive power, connections to logical inference, close links to syntax, and potential for constraint-based disambiguation (Copestake et al., 2005). We develop methods for benchmarking MRS approaches for dialogue based on annotations expressed in terms of Abstract Meaning Representations (AMR), by comparing the consistency of predicate-argument relations across representations, thus showing that MRS shows promise for TOD. Our evaluation shows that in 94.1% of cases, our implementation of MRS using spaCy yields edges consistent with gold-standard predicate-argument relations annotated in a cooking domain (Jiang et al., 2022).

2 Related Work

2.1 LLM TOD systems

Multiple recent TOD systems have been built using LLMs and specialized NLU modules for their specific task. However, most end-to-end LLMs can struggle in three areas. The first is with lookahead planning problems, where understanding the final goal is crucial to avoid early errors that can obstruct later steps. Bachmann and Nagarajan (2024) demonstrate cases where models trained to solve problems using only next-token prediction struggle to learn what the model should choose for the first token. Momennejad et al. (2023) and Valmeekam et al. (2023) found that models struggle on planning tasks framed as word problems. The second area

Proceedings of the 6th International Workshop on Designing Meaning Representations, pages 30–37

where LLMs may struggle is reasoning. Jiang et al. (2024) determined that state-of-the-art LLMs fail to reason consistently across minor variations, such as changing names of people or places. The third area is belief state tracking, where it has been seen that an end-to-end LLM inference compares poorly to supervised models. Hudeček and Dusek (2023) shows five state-of-the-art models performing better than LLMs, 3 of which use an NLU component or a semantic representation (Feng et al., 2021; Sun et al., 2023; Zhu et al., 2023).

LLM-based systems can stage multiple prompts to perform dialogue state tracking, knowledge retrieval, and dialogue planning (Dong et al., 2025; Xu et al., 2024; Zhang et al., 2023). However, as the amount of LLM calls or tokens in the output increase, the inference latency of LLMs can become a pain point for real-time dialogue systems; many AI assistants require a response within a particular time frame, such as Alexa's 8-second requirement for responses.[1] The specialized components that TOD systems use to achieve real-time performance—track belief states (Hudeček and Dusek, 2023) or generate responses(Chiu et al., 2023)—typically rely on explicit semantic representations.

2.2 TOD systems using Procedural Semantic Representations

One approach to explicit semantics in TOD is procedural semantics (Bollini et al., 2013; Nevens et al., 2024; Verheyen et al., 2023). Procedural semantics offers representations for task descriptions that are specific enough to be executed programmatically and achieve desired results. Ultimately, collaborative agents need executable action representations, but there are potential disadvantages to deriving those representations directly from utterances. Deriving them may involve planning and plan recognition as well as processes of compositional interpretation and resolution of grounded references (Geib et al., 2022). For example, action plans may depend on the capabilities of the agent and the physical state of the environment. An abstract semantic representation can play an important role for collaborative dialogue by representing task content in a way that can be shared across agents and contexts and can mediate between various kinds of linguistic and plan-based reasoning.

[1] https://developer.amazon.com/docs/alexa/custom-skills/send-the-user-a-progressive-response.html

2.3 TOD systems using AMR

AMR is another form of semantic representation used in NLU modules for TOD (Tam et al., 2023). AMR represents each sentence as a rooted, directed, acyclic graph. In the graph, each edge has a label for the relation, and each leaf represents a concept (Banarescu et al., 2013). These graphs can also be written in PENMAN notation (Matthiessen and Bateman, 1992). AMR has been extended to be more suitable for representing dialogues (O'Gorman et al., 2018; Bonial et al., 2020) and multimodal communication (Brutti et al., 2022). Tam et al. (2023) has shown that AMR can be used to annotate actions for both human-human interactions and human-object interactions. AMR has also shown promise in TOD through interactive simulations (Krishnaswamy et al., 2017).

2.4 Minimal Recursion Semantics

We have chosen to use MRS in our work. MRS is a framework that can encode predicate arguments and other grammatical constraints on lexical and phrasal semantics to generate flat semantic representations. An MRS structure is a tuple containing a top handle (GT), a bag of elementary predicates or EPs ("an EP is a single relation with its associated arguments"), and a bag of handle constraints (C) (Copestake et al., 2005). Like AMR, MRS is scalable because it abstracts away from domain-specific content.

While AMR is easy to annotate, and has become a popular semantic representation for text-based tasks, AMR does not support constraint-based ambiguity resolution like MRS does (Copestake et al., 2005; Wein, 2025). The incremental constraint-based approach of MRS also streamlines the representation of dialogue processes such as clarification, thereby facilitating system efforts to ensure common ground. In addition, AMR lacks the full logical expressiveness of MRS (Bender et al., 2015; Bos, 2016), which underpins logical approaches to bridging semantic and common-sense inferences (Hobbs, 1985; Copestake et al., 2005).

We have chosen not to build on existing MRS implementations, such as English Resource Grammar (ERG) (Flickinger et al., 2000) [2], because our approach allows for more flexibility, such as choosing to ignore scopal arguments (which would not have an impact when combining linguistic reasoning and plan-based inferences, since planning modules typ-

[2] https://delph-in.github.io/delphin-viz/demo/

ically do not account for scopal arguments), therefore allowing for a more lightweight and efficient representation. Our MRS implementation is builds on dependency parsing provided by spaCy. This decision is primarily for convenience; dependencies provide a simple and effective starting point for our work. We believe our approach could be adapted as needed to other state-of-the-art real-time dependency or constituency parsers.

3 System Design

For this paper, MRS is used as an early component of NLU to help create a logical form (LF) as a semantic representation that can be used for dialogue systems and updated with information from the planner's inferences, allowing the LF to be updated with information from the environment. Since we are comparing MRS to AMR (to show that if AMR is used in TOD, MRS should be able to do so as well), we will focus on non-scopal EPs, ignoring all EPs that can be a scopal EP (such as adverbs).[3]

3.1 spaCy

For dependency parsing, spaCy was selected due to its popularity and its capability for real-time dependency parsing. It is a transition-based dependency parser that uses an arc-eager system. SpaCy's English models were trained using OntoNotes 5.0 (Weischedel et al., 2013), which contains approximately 1.5 million words from news media, telephone conversations, broadcast conversations, and weblogs. SpaCy's developers report a 95.1% accuracy for unlabeled attachment score (UAS) and 93.7% labeled attachment score (LAS) accuracy when tested on the Penn Treebank (Marcus et al., 1993)[4], which contains articles from the Wall Street Journal (WSJ) from 1984 to 1989. However, a machine learning model evaluated on WSJ may have different accuracy for other domains. We took Cookdial and evaluated predicate-argument relations reported by spaCy and translated to MRS (discussed in Section 4) to determine their consistency with the corresponding Extended-AMR (EAMR). We used spaCy version 3.7.4 with en_core_web_lg model version 3.7.1.

3.2 Implementation

Algorithm 1 shows the logic used to implement MRS to create an LF. It assumes that each word in the sentence is associated with a head, a dependency label, a part-of-speech (POS) tag, and its position in the sentence. Each word may also be associated with a predicate (the meaning carried by the word) and a variable (the discourse referent it evokes).

Algorithm 1 Build MRS LF from Dependencies

Input: *sent* = sentence
Output: *lf*
lf = set()
$ignore\_deps$ ={det,punct,case,adv}
for all $(child, rel, head) \in sent.deps()$ **do**
  if $is\_pred(child)$ **then**
    lf.add $([child.pred, child.var])$
    if $rel \in$ UD_Modifiers **then**
      lf.add$([=, head.var, child.var])$
    if $child.tag =$ VBG **then**
      lf.add$([$nsubj$, child.var, head.var])$
    else if $child.tag =$ VBN **then**
      lf.add$([$dobj$, child.var, head.var])$
  if $rel =$ pobj, dobj **then**
    lf.add$([role(rel, head),$
    $head.head.var, child.var])$
  else if $rel \notin ignore\_deps$ **then**
    lf.add$([rel, head.var, child.var])$
return *lf*

The algorithm loops through each relation in the sentence, focusing on representing the contribution of the dependent element (child). Nouns, pronouns, adjectives, verbs, and auxiliaries without dependents contribute elementary predications. Verbal dependent modifiers assign an appropriate syntactic role to the head referent (subject for present participle, object for past participle). All other modifiers, excluding adverb modifiers which are ignored, equate their variable to the variable of the elementary predicate they are describing. Objects of prepositions are assigned a suitable semantic role with respect to the entity modified by the preposition. Aside from root, determiners, punctuation, adverbs, and case modifiers, all other dependency labels are included in the logical form. Since planning modules typically do not account for scopal arguments, determiners and adverb modifiers have been excluded from consideration.

For the sentence "Pour cranberry juice into a 5-cup ring mold", the MRS algorithm will go through each relation given by spaCy (as shown in Figure 1). If the first dependency identified is the direct object

[3] Note that an entire MRS structure can generally be created with a dependency tree parse.

[4] https://spacy.io/usage/facts-figures

Token	Relation	Part of Speech	Tag	Head	Children	Ancestors
Pour	root	VERB	VB	Pour	[juice,into]	[ ]
cranberry	compound	NOUN	NN	juice	[]	[juice, Pour]
juice	dobj	NOUN	NN	Pour	[cranberry]	[Pour]
into	prep	ADP	IN	Pour	[mold]	[Pour]
a	det	DET	DT	mold	[]	[mold, into, Pour]
5	nummod	NUM	CD	cup	[ ]	[cup, mold, into, Pour]
-	punct	PUNCT	HYPH	cup	[ ]	[cup, mold, into, Pour]
cup	compound	NOUN	NN	mold	[5, -]	[mold, into, Pour]
ring	compound	NOUN	NN	mold	[ ]	[mold, into, Pour]
mold	pobj	NOUN	NN	into	[a, cup, ring]	[into, Pour]
.	punct	PUNCT	.	Pour	[ ]	[Pour]

Table 1: spaCy parse of "Pour cranberry juice into a 5-cup ring mold."

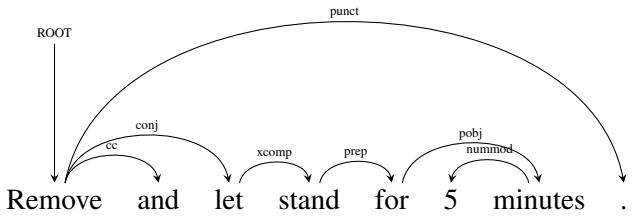

Figure 1: spaCy dependency parse of the sentence: “Remove and let stand for 5 minutes.” Parsed using spaCy.

relationship between the head "Pour" and the child "juice", the algorithm identifies that "juice" evokes a discourse referent, and stores the fact that the predicate "juice" applies to the referent "x_juice_2" by storing [juice, "x_juice_2"] Then, it will identify that the dependency is not a Universal Dependency modifier, and that the dependency is not a preposition, so it will be represented as ["dobj", x_Pour_0, x_juice_2]. This process will be completed while going through all remaining dependencies.

```
(inst-0 / R
  :inform (ac-0-0 "Pour" 3:7/ AC
    :ppt (ing-0 "cranberry juice" 8:23 / FOOD)
    :gol(tool-0-0 "a 5-cup ring mold" 29:46 / TOOL)
    :_result (juice-in-mold))
```

Figure 2: EAMR representation of the instruction “Pour cranberry juice into a 5-cup ring mold.”

For the sentence "Remove and let stand for 5 minutes.", the MRS algorithm will go through each relation given by spaCy (as shown in Figure 4). It will identify and store the elementary predications, "remove", "let", "stand", "5", "minutes" as before. For example, "remove" will be stored as [remove, "x_remove_0"]. It will stores additional relations such as noting the nummod relation between "minutes" and "5" as [’=’, ’x_5_5’, ’x_minutes_6’]. The remaining relations are from the else if clause on line 17. These relations are: [’cc’, ’x_Remove_0’, ’and’], [’conj’, ’x_Remove_0’, ’x_let_2’], [’xcomp’, ’x_let_2’, ’x_stand_3’], [’for’, ’x_stand_3’, ’x_minutes_6’]. When supplied to reference resolution and clarification module, we can potentially recognize that "remove" and "stand" concern an implicit object derived from dialogue context. When combined with a planner module, the planner could infer how to achieve the successive "remove" and "stand" tasks with suitable planner actions.

```
(inst-8 / R
  :inform (ac-8-0 "Remove" 3:9/ AC
    :ppt (NULL / FOOD)
    :ppt (NULL / FOOD)
  :inform (ac-8-0 "stand" 18:23/ AC
    :duration (dur-8-0 "5 minutes"@28:37 / DUR)
    :ppt (NULL / FOOD)))
```

Figure 3: EAMR representation of the instruction “Remove and let stand for 5 minutes.”

4 Evaluation

For our evaluation we chose the Cookdial dataset (Jiang et al., 2022). The data set contains Extended-AMR (EAMR) annotations of recipe instructions, which mimic many ideas and notations from AMR

(Jiang et al., 2022). EAMR uses PENMAN notation (the string and index annotations are placed into ":name" or ":named"), and represents a directed acyclic graph composed of nodes (the entity type) and edges (relation between the predicate and its arguments) (Jiang et al., 2022). For the purposes of evaluation, we will be only considering EAMR with multiple edges or nodes, since EAMR of just a single node would not have any significant information to compare against spaCy's parse, as the entire sentence would be the constituent. This provides us with 227 sentences, totaling 951 constituents to evaluate for the consistency of predicate-argument relations in EAMR captured in both the spaCy parse and MRS clauses.

4.1 Predicate-Argument Consistency

We recursively iterate through the AMR graph, starting from its root node (Algorithm 2 in Appendix), and verify if each constituent has exactly one semantic relation with a different constituent (Algorithm 3 in Appendix). This is done by identifying and counting the external semantic relations the constituent has, and by verifying the alignment of AMR with the dependency head relation (that would be provided to MRS) by spaCy's parse. For example, if you consider Figure 2, the phrase "cranberry juice", we would confirm that there is only one external semantic relation, which in this case would be the head verb "Pour". This means no additional dependencies link to a word in the phrase from elsewhere in the AMR graph, therefore showing that the EAMR and spaCy parse are consistent. This evaluation can be applied across any AMR that contains multiple edges or nodes by following the same methods.

4.2 Evaluation Results

Out of the 951 edges evaluated, it was found that 56 had inconsistent constituency ($\approx 5.9\%$). This performance ($\approx 94.1\%$) is comparable with spaCy RoBERTa (2020) dependency parsing accuracy on Penn Treebank (Marcus et al., 1993), which is 95.1% for unlabeled attachment score.[5] Note that spaCy had incorrectly interpreted "in." as the end of a sentence for two utterances; therefore, it was decided "inch" would be substituted for "in." While this analysis of the consistency of the Dependency Parser's and MRS algorithm highlights specific limitations of the parser, the implications of the dependency parser's accuracy for the LF are not yet fully understood.

[5] https://spacy.io/usage/facts-figures

5 Conclusion

In this paper, we have built on existing AMR annotations to argue that MRS may also be used for semantic representations in TOD. We showed how to evaluate MRS by comparing predicate-argument relations in the input of MRS to those annotated in EAMR for a cooking domain. Evaluation shows that MRS aligns with EAMR relations with 94.1% accuracy when using spaCy's dependency parsing as the main input for our MRS algorithm.

In future work, we plan to explore further uses of MRS as structured, semantic representations to bridge language-based and plan-based inferences for TOD. We hope to develop a versatile NLU module that can be used across multiple domains and even languages—since the Universal Dependencies framework provides consistent cross-linguistic grammar annotations (de Marneffe et al., 2021). We further hope to build on strategies from Traum (1995) and Rich et al. (2001) to allow for tracking and maintaining common ground in collaborative interactions. Finally, we are interested in using our MRS module for coordinating activity by extending our existing implementation of plan filtering and semantic grounding using planning and plan recognition (Geib et al., 2022).

Limitations

While this paper evaluates the dependency parse on EAMR, only relations between EAMR nodes are tracked, leaving out node-internal relations, such as the relation between "cranberry" and "juice" in the EAMR constituent "cranberry juice". Also, while our NLU module may be applicable across domains, it will still require planning modules that may have to be created for each domain, as well as a knowledge base for each domain to identify action types and resolve references. We have also not demonstrated the impact of our techniques on dialogue quality or task success.

Acknowledgments

Thanks to Rich Magnotti and the reviewers for helpful feedback. Supported by NSF awards 2021628, 2119265, and 2427646.

References

Gregor Bachmann and Vaishnavh Nagarajan. 2024. The pitfalls of next-token prediction. *Preprint*, arXiv:2403.06963.

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.

Emily M. Bender, Dan Flickinger, Stephan Oepen, Woodley Packard, and Ann Copestake. 2015. Layers of interpretation: On grammar and compositionality. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 239–249, London, UK. Association for Computational Linguistics.

Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. 2013. Interpreting and executing recipes with a cooking robot. In *Experimental Robotics: The 13th International Symposium on Experimental Robotics*, pages 481–495. Springer.

Claire Bonial, Lucia Donatelli, Mitchell Abrams, Stephanie M. Lukin, Stephen Tratz, Matthew Marge, Ron Artstein, David Traum, and Clare Voss. 2020. Dialogue-AMR: Abstract Meaning Representation for dialogue. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 684–695, Marseille, France. European Language Resources Association.

Johan Bos. 2016. Squib: Expressive power of Abstract Meaning Representations. *Computational Linguistics*, 42(3):527–535.

Richard Brutti, Lucia Donatelli, Kenneth Lai, and James Pustejovsky. 2022. Abstract Meaning Representation for gesture. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1576–1583, Marseille, France. European Language Resources Association.

Justin Chiu, Wenting Zhao, Derek Chen, Saujas Vaduguru, Alexander Rush, and Daniel Fried. 2023. Symbolic planning and code generation for grounded dialogue. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7426–7436, Singapore. Association for Computational Linguistics.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan Sag. 2005. Minimal recursion semantics: An introduction. *Reseach On Language And Computation*, 3:281–332.

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

Wenjie Dong, Sirong Chen, and Yan Yang. 2025. ProTOD: Proactive task-oriented dialogue system based on large language model. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9147–9164, Abu Dhabi, UAE. Association for Computational Linguistics.

Yue Feng, Yang Wang, and Hang Li. 2021. A sequence-to-sequence approach to dialogue state tracking. 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 1714–1725, Online. Association for Computational Linguistics.

Dan Flickinger, Ann Copestake, and Ivan A. Sag. 2000. *HPSG Analysis of English*, pages 254–263. Springer Berlin Heidelberg, Berlin, Heidelberg.

Christopher Geib, Denson George, Baber Khalid, Richard Magnotti, and Matthew Stone. 2022. An integrated architecture for common ground in collaboration. *ACS 2022*.

Jerry R. Hobbs. 1985. Ontological promiscuity. In *23rd Annual Meeting of the Association for Computational Linguistics*, pages 60–69, Chicago, Illinois, USA. Association for Computational Linguistics.

Vojtěch Hudeček and Ondrej Dusek. 2023. Are large language models all you need for task-oriented dialogue? In *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 216–228, Prague, Czechia. Association for Computational Linguistics.

Bowen Jiang, Yangxinyu Xie, Zhuoqun Hao, Xiaomeng Wang, Tanwi Mallick, Weijie J Su, Camillo Jose Taylor, and Dan Roth. 2024. A peek into token bias: Large language models are not yet genuine reasoners. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4722–4756, Miami, Florida, USA. Association for Computational Linguistics.

Yiwei Jiang, Klim Zaporojets, Johannes Deleu, Thomas Demeester, and Chris Develder. 2022. Cookdial: a dataset for task-oriented dialogs grounded in procedural documents. *Applied Intelligence*, 53(4):4748–4766.

Nikhil Krishnaswamy, Pradyumna Narayana, Isaac Wang, Kyeongmin Rim, Rahul Bangar, Dhruva Patil, Gururaj Mulay, Ross Beveridge, Jaime Ruiz, Bruce Draper, and James Pustejovsky. 2017. Communicating and acting: Understanding gesture in simulation semantics. In *Proceedings of the 12th International Conference on Computational Semantics (IWCS) — Short papers*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Christian M.I.M. Matthiessen and John A. Bateman. 1992. Text generation and systemic-functional linguistics: Experiences from english and japanese.

Ida Momennejad, Hosein Hasanbeig, Felipe Vieira, Hiteshi Sharma, Robert Osazuwa Ness, Nebojsa Jojic, Hamid Palangi, and Jonathan Larson. 2023. Evaluating cognitive maps and planning in large language models with cogeval. *Preprint*, arXiv:2309.15129.

Jens Nevens, Robin De Haes, Rachel Ringe, Mihai Pomarlan, Robert Porzel, Katrien Beuls, and Paul Van Eecke. 2024. A benchmark for recipe understanding in artificial agents. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation*, pages 22–42.

Tim O'Gorman, Michael Regan, Kira Griffitt, Ulf Hermjakob, Kevin Knight, and Martha Palmer. 2018. AMR beyond the sentence: the multi-sentence AMR corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3693–3702, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

OpenAI. 2022. Introducing ChatGPT. `https://openai.com/blog/chatgpt`.

OpenAI. 2024. GPT-4 technical report. *Preprint*, arXiv:2303.08774.

Charles Rich, Candace L Sidner, and Neal Lesh. 2001. Collagen: Applying collaborative discourse theory to human-computer interaction. *AI magazine*, 22(4):15–15.

Haipeng Sun, Junwei Bao, Youzheng Wu, and Xiaodong He. 2023. Mars: Modeling context state representations with contrastive learning for end-to-end task-oriented dialog. *Preprint*, arXiv:2210.08917.

Christopher Tam, Richard Brutti, Kenneth Lai, and James Pustejovsky. 2023. Annotating situated actions in dialogue. In *Proceedings of the Fourth International Workshop on Designing Meaning Representations*, pages 45–51, Nancy, France. Association for Computational Linguistics.

David Rood Traum. 1995. *A computational theory of grounding in natural language conversation*. University of Rochester.

Karthik Valmeekam, Matthew Marquez, and Subbarao Kambhampati. 2023. Can large language models really improve by self-critiquing their own plans? *Preprint*, arXiv:2310.08118.

Lara Verheyen, Jérôme Botoko Ekila, Jens Nevens, Paul Van Eecke, and Katrien Beuls. 2023. Neuro-symbolic procedural semantics for reasoning-intensive visual dialogue tasks. In *ECAI 2023*, pages 2419–2426. IOS Press.

Shira Wein. 2025. Ambiguity and disagreement in Abstract Meaning Representation. In *Proceedings of Context and Meaning: Navigating Disagreements in NLP Annotation*, pages 145–154, Abu Dhabi, UAE. International Committee on Computational Linguistics.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. OntoNotes release 5.0. *Preprint*, Linguistic Data Consortium:2303.08774.

Heng-Da Xu, Xian-Ling Mao, Puhai Yang, Fanshu Sun, and Heyan Huang. 2024. Rethinking task-oriented dialogue systems: From complex modularity to zero-shot autonomous agent. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2748–2763, Bangkok, Thailand. Association for Computational Linguistics.

Xiaoying Zhang, Baolin Peng, Kun Li, Jingyan Zhou, and Helen Meng. 2023. Sgp-tod: Building task bots effortlessly via schema-guided llm prompting. *Preprint*, arXiv:2305.09067.

Qi Zhu, Christian Geishauser, Hsien chin Lin, Carel van Niekerk, Baolin Peng, Zheng Zhang, Michael Heck, Nurul Lubis, Dazhen Wan, Xiaochen Zhu, Jianfeng Gao, Milica Gašić, and Minlie Huang. 2023. Convlab-3: A flexible dialogue system toolkit based on a unified data format. *Preprint*, arXiv:2211.17148.

A Appendix

A.1 Algorithms

We present Algorithm 2 to show how the AMR graph was traversed while checking relations. We gave each node in a sentence a unique identification, and for each relation in the AMR, we would call Algorithm 3, and report the returned results.

In Algorithm 3, we show how we verify if each constituent has exactly one semantic relation with a different constituent, and how we verify the alignment of the AMR graph and spaCy's parse.

A.2 spaCy Dependency Diagram

Table 1 presents the spaCy dependency parse for the example sentence "Pour cranberry juice into a 5-cup ring mold".

Algorithm 2 AMR_TRAVERSAL

Input: *node_id* = first node id in *amr_graph*, *amr_graph*, *visited* = [], *prev_word=None*
Output: Dataframe updated by *report_result* function
if *node_id* in *visited* **then return**
visited.add(*node_id*)
head_node = *amr_graph*[*node_id*]
for each *child_id* in *head_node.relations* **do**
    child_node = *amr_graph*[*child_id*]
    if *prev_word* $\neq$ *None* **then**
        relations, *head_relations* = *check_relation*(*head_node.words*, *child_node.words*)
        report_result(*relations*, *head_relations*)
    Traverse_AMR(*child_id*, *amr_graph*, *visited*, *node_id*)

Algorithm 3 CHECK_RELATION(WORDS, HEAD_WORDS)

Input: *words*, *head_words*
Output: *relations*, *head_relations*
relations = []
head_relations = []
apart_relations = []
for each *word* in *words*: **do**
    if *word.head* not in *words* **then**
        relations.append(*word.head*)

if len(*relations*) == 1 : **then**
    for *word* in *head_words* : **do**
        if *word* in *relations* **then**
            head_relations.append(*word*)
        else
            ancestors = get_ancestors(*words*, *word*)
            for *ancestor* in *ancestors* **do**
                if *ancestor* == *word and* not in *apart_relation* **and** not in *words* **then**
                    apart_relation.append(*ancestor*)
    if len(*head_relations*) < 1: **then**
        head_relations = *apart_relation*
return *relations*, *head_relations*

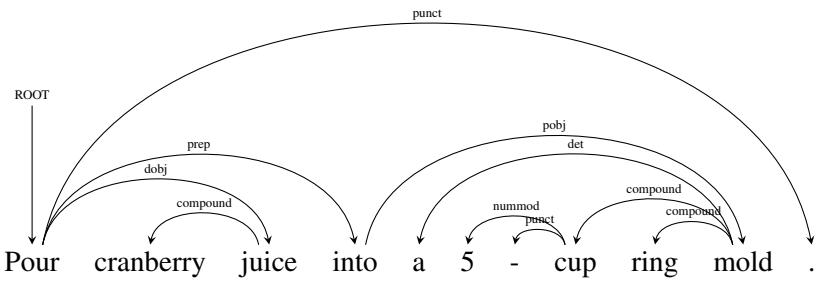

Figure 4: spaCy dependency parse of the sentence: “Pour cranberry juice into a 5-cup ring mold.” Parsed using spaCy.