Yanming Li, Meaghan Fowlie

GPT makes a poor AMR parser

Abstract

This paper evaluates GPT models as out-of-the-box Abstract Meaning Representation (AMR) parsers using prompt-based strategies, including 0-shot, few-shot, Chain-of-Thought (CoT), and a two-step approach in which core arguments and non-core roles are handled separately. Our results show that GPT-3.5 and GPT-4o fall well short of state-of-the-art parsers, with a maximum Smatch score of 60 using GPT-4o in a 5-shot setting. While CoT prompting provides some interpretability, it does not improve performance. We further conduct fine-grained evaluations, revealing GPT's limited ability to handle AMR-specific linguistic structures and complex semantic roles. Our findings suggest that, despite recent advances, GPT models are not yet suitable as standalone AMR parsers.

1 Introduction

Much of Abstract Meaning Representation (AMR) parsing is currently concentrated on fine-tuning pre-trained language models such as BART (Lewis et al., 2020). Newer Large Language Models (LLMs) such as GPT bring a new paradigm for NLP research: prompting. LLMs also show impressive "reasoning" capabilities and a certain kind of interpretability with Chain-of-Thought (CoT) prompting. With prompt-based learning, an LLM might be capable of just about any NLP task, if the right prompt and mapping from output text to task output can be found (P. Liu et al., 2023).

This paper explores the possibility that AMR parsing is possible if the requested output is in PENMAN notation. In this paper, we apply a variety of prompting strategies to induce GPT to do AMR parsing. We demonstrate that GPT models are insufficient as AMR parsers. Our work also results in two main findings. First, Chain-of-Thought prompting is also ineffective, though it offers analytical insights and shows some potential. Second, decomposing the task into identifying core argument roles and modifiers did not improve performance. Beyond these findings, we contribute an additional fine-grained evaluation for deeper analysis.

2 Background & Related Work

Abstract Meaning Representation An AMR (Figure 1) is composed of labelled nodes and edges, where nodes represent *concepts* – roughly the words or semantic units of the sentence – and edges represent the relationships between them. Formally, an AMR graph can be expressed as a set of triples (s, r, t), where s is the source concept (head),

r is a semantic relation label (e.g. `:ARG0`, `:mod`), and t is the target concept or value (Goodman, 2020).

AMR guidelines[1] specify details such as the use of PropBank (Choi, Bonial, & Palmer, 2010) verb senses (e.g. `receive-01`) and numbered arguments (e.g. `:ARG0`), named entity subgraphs, and negation, indicated with `(:polarity -)`. The numerical suffix in `receive-01` denotes a verb sense (here: `get something`), while `:ARG0` typically denotes the receiver. AMRs are written in Penman notation, a parenthesis-based representation for nested graphs, which allows text-based models to generate them directly (van Noord & Bos, 2017).

AMR Parsing is the task of generating an AMR given a sentence. Existing AMR parsers mainly fall into three categories: transition-based models, sequence-to-graph models, and sequence-to-sequence (seq2seq) models. Transition-based models generate new nodes, edges, or subgraphs based on the words of the sentence (Fernandez Astudillo, Ballesteros, Naseem, Blodgett, & Florian, 2020; Lindemann, Groschwitz, & Koller, 2020; Naseem et al., 2019; Peng, Gildea, & Satta, 2018; Zhou, Naseem, Fernandez Astudillo, & Florian, 2021). Sequence-to-graph models derive the graph from existing nodes without transition processes, directly extending new nodes and edges (D. Cai & Lam, 2020; Zhang, Ma, Duh, & Van Durme, 2019). Seq2seq models directly generate the text format of AMRs from raw sentences (Bai, Chen, & Zhang, 2022; Blloshmi, Tripodi, & Navigli, 2020; Lee et al., 2022; van Noord & Bos, 2017; Vasylenko, Huguet Cabot, Martínez Lorenzo, & Navigli, 2023). We use GPT as a seq2seq model.

A parallel study by Ettinger, Hwang, Pyatkin, Bhagavatula, and Choi (2023) also investigates AMR parsing with GPT models, using similar prompting strategies such as 0-shot and 5-shot prompting. While our results are slightly better, both studies remain far from state-of-the-art performance. Compared to their work, our experiments are conducted on larger datasets, include novel prompting strategies, and provide fine-grained analysis using GrAPES (Groschwitz, Cohen, Donatelli, & Fowlie, 2023). We also systematically evaluate the GPT model's ability to generate well-formed AMRs (termed Parsability), and show that post-processing significantly improves Parsability to over 90% in all settings except 0-shot.

In-context learning/k-shot/few-shot prompting is a gradient-free "learning" strategy for language models that provides k task-related example question-answer pairs before asking the target question (Brown et al., 2020; Dong et al., 2024; Wei, Tay, et al., 2022). Few-shot prompting generally has better performance than 0-shot prompting (J. Liu et al., 2022; Min et al., 2022; Zhao, Wallace, Feng, Klein, & Singh, 2021), which only provides instructions. Performance is sensitive to the prompt, including the number of shots (Cao, Law, & Fidler, 2020) and the choice of examples (Zhao et al., 2021).

[1] https://github.com/amrisi/amr-guidelines/blob/master/amr.md

Chain-of-Thought (CoT) Prompting Unlike regular few-shot prompting, CoT prompts include not only example question-answer pairs but also intermediate reasoning steps that can derive the final answer (Wei, Wang, et al., 2022). CoT prompting can significantly enhance the capabilities of LLMs in complex reasoning (Lewkowycz et al., 2022; Saparov & He, 2023), and bring more interpretability with the generated reasoning process (Weng et al., 2023). Madaan and Yazdanbakhsh (2022) for instance claim that, through CoT prompting, LLMs can better understand the task by extracting commonsense knowledge from the questions, and generalize to unseen tasks by mimicking the expert's intermediate reasoning steps (Yang, Schuurmans, Abbeel, & Nachum, 2022). However, final answers can be inconsistent with reasoning steps (Lyu et al., 2023).

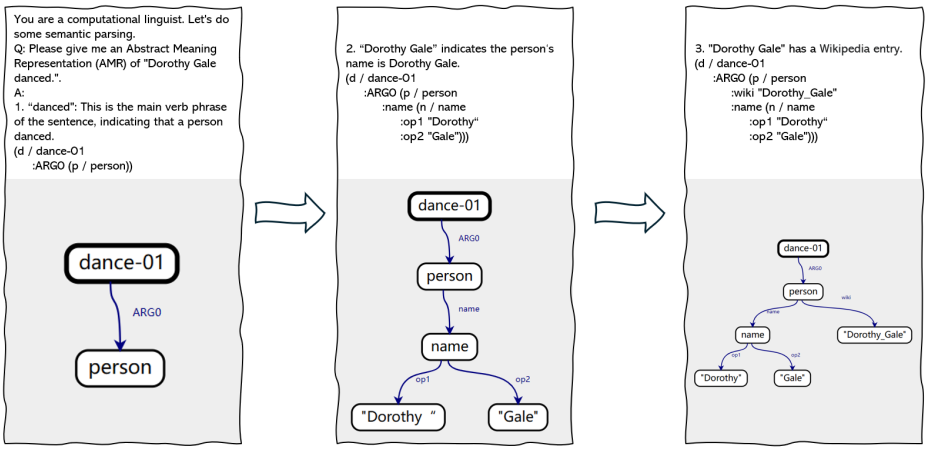


Figure 1: A toy example of CoT reasoning (Top-down) for *Dorothy Gale danced.* Each box is a reasoning step, where the top part is the CoT reasoning text and the bottom part is a visualization of the corresponding subgraph

3 Experimental Setup and Prompting Strategies

In this paper, we evaluate multiple prompting strategies for generating AMRs using GPT models, including three GPT-3.5 variants—`text-davinci-003`, `gpt-3.5-turbo`, and `gpt-3.5-turbo-instruct` (Brown et al., 2020)—as well as GPT-4o (OpenAI et al., 2024). All experiments are conducted via the official OpenAI API, with temperature set to 0 for reproducibility.[2]

[2]The `gpt-3.5-turbo-instruct` and `GPT-4o` models used correspond to `gpt-3.5-turbo-0613` and `gpt-4o-2024-05-13` at the time of our experiments. Used within terms of use: https://openai.com/policies/eu-terms-of-use/

3.1 AMR Dataset

Our experiment was conducted on the English AMR 2.0 (Knight et al., 2017) and AMR 3.0 (Knight et al., 2021) test set with example selections for few-shot prompting on the training set of AMR 2.0 and AMR 3.0, respectively. AMR 2.0 test set has 1,371 AMRs, (test) and 36,521 (train) AMRs, and AMR 3.0 has 1,898 and 55,635. The AMR 2.0 test set is essentially a large subset of that of AMR 3.0. Used within the terms of the license, LDC User Agreement for Non-Members.

3.2 Prompting Strategies

All of our prompts request an AMR given a sentence; some include examples. Since role-playing improves model performance (Kong et al., 2023; Reynolds & McDonell, 2021), all prompts begin with *"You are a computational linguist."* We implement and evaluate five prompting strategies: 0-shot, 1-shot, 5-shot, Chain-of-Thought (CoT) with one example, and two-step prompting (details in Appendix A).

1-shot and CoT contain a predefined example sentence: *"The poor kid didn't receive the gift and the postcard that Dorothy Gale sent him on May 25th."*, which was built to demonstrate common AMR properties, such as reentrancy, different non-core roles, etc.

5-shot examples are sampled from the training set using two strategies: random sampling and semantic similarity-based sampling. Using semantically similar examples in prompts can improve LLM performance (Gao, Fisch, & Chen, 2021; J. Liu et al., 2022). We employ Wang et al. (2020)'s model *"sentence-transformers/all-MiniLM-L6-v2"*[3] to compute cosine similarities between the target sentence and training sentences, selecting the top five most similar examples for the prompt.

We introduce two different styles of CoT prompts, top-down (see Figure 1) and bottom-up. The top-down approach begins by identifying the top node, typically the main verb, and subsequently determines its child nodes and their semantic relations in a recursive manner until the complete graph is constructed. The bottom-up approach initially extracts smaller subgraphs, such as the subject, object, location, and time, and then incrementally links these subgraphs through their interrelations until the entire graph is assembled. (See Appendix A.4 for a bottom-up example.)

The two-step prompting strategy combines elements of Chain-of-Thought (CoT) and 5-shot prompting. In Step 1, the model is prompted using 5-shot examples to generate only the core arguments (The nodes linked by labels such as ARG0, ARG1, etc.). In Step 2, a new GPT instance receives the output from Step 1 along with a list of AMR non-core roles (e.g., modifiers) and is prompted to incrementally add non-core roles, guided by a single CoT example.

[3] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

4 Results

Parsability of outputs: Some LLM output texts are not parsable as AMRs, and some produce multiple AMRs. While the ability of the model to produce a single, valid AMR is the main question of interest, also of interest is whether a pipeline that includes minor post-processing works as an AMR parser. Thus we also apply a post-processing script[4] that fixes mismatched parentheses, splits multiply-labeled nodes into separate nodes, and combines multiple AMRs into one. The proportion of valid AMRs for each method (termed Parsability), before and after post-processing, is in Figure 2.

Without post-processing, in the 0-shot scenario, fully 85% of outputs from GPT -3.5 had syntactic errors rendering them unparsable, and post-processing only brought the parsability rate up to 33%. At the other end of the scale, GPT-4o in the 5-shot scenario was able to generate 86% parsable outputs, and post-processing brought it up to 98%. Thus, left to its own devices, GPT-3.5 at least is entirely unusable, but the more advanced GPT-4o, plus post-processing, is able to generate AMR-formatted text given 5 examples.

Smatch: The standard evaluation metric for AMR is **Smatch** (S. Cai & Knight, 2013), which computes the F1 score over the best alignment of triples between the predicted and gold AMR graphs. Each AMR is represented as a set of triples (s, r, t), where s and t are concepts and r is a relation. Smatch is defined as:

$$\text{Smatch} = \frac{2 \times |G_p \cap G_g|}{|G_p| + |G_g|}$$

where G_p and G_g are the sets of triples in the predicted and gold graphs. Here, unparsable graphs were replaced by a dummy graph `(d / dummy)`.

Table 1 shows the Smatch scores of the best version of each method after post-processing, compared with the SOTA AMR parser. Full results, including scores before and after post-processing, are provided in Appendix B.1.

Our best results are for the 5-shot method with GPT-4o (Smatch 60), but nothing approaches the SOTA AMR parser (Vasylenko et al., 2023) with 86.1 on AMR 2.0 and with 84.6 on AMR 3.0. GPT is loosely comparable to an early AMR baseline parser, JAMR, with a 58 Smatch on the original LDC2013E117 AMR dataset (2,100 test sentences; Flanigan, Thomson, Carbonell, Dyer, and Smith (2014)).

The 1-shot and CoT methods provide only one example, and perform poorly, with CoT actually worsening performance (Smatch 36 *vs* 41 with GPT-3.5). The performance of the two-step method was about the same as 5-shot (49 and 50 with GPT-3.5).

Our Smatch results are in keeping with parallel work done by Ettinger et al. (2023), who find that, at best, GPT outputs on the standard AMR 3.0 test set have a Smatch score of around 50.

[4]Script can be found in `https://github.com/liam-0/Fix-ill-formed-AMR.git`

Hand-analysis of CoT sample outputs on AMR 3.0 (Appendix C.1) found a myriad of errors, including mismatches between the reasoning step and partial result. Still, the sampled CoT outputs were, to us, surprisingly good, often making sense and matching the subgraphs generated.

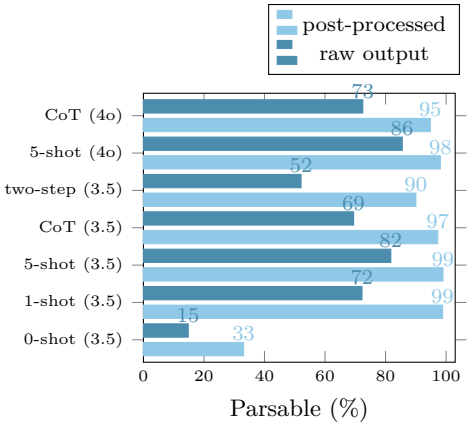


Figure 2: Parsability before/after post-processing. The GPT version is indicated in parentheses after each prompting strategy (e.g., CoT (4o))

Dataset AMR 3.0	**Smatch**
5-shot (GPT-4o)	60
CoT (GPT-4o)	55
LeakDistill (SOTA) (Vasylenko et al., 2023)	**84.6**
Dataset AMR 2.0	
0-shot (GPT-3.5)	14
1-shot (GPT-3.5)	41
5-shot (GPT-3.5)	50
CoT (GPT-3.5)	36
two-step (GPT-3.5)	49
LeakDistill (SOTA) (Vasylenko et al., 2023)	**86.1**
Dataset LDC2013E117	
JAMR (Flanigan et al., 2014)	58

Table 1: Smatch for our methods (all post-processed), LeakDistill (SOTA), and JAMR (an early baseline).

4.1 Fine-grained results

In addition to Smatch, we evaluated the GPT-4o outputs with the Granular AMR Parsing Evaluation Suite, or GrAPES (Groschwitz et al., 2023), a fine-grained evaluation with 36 categories divided into 9 sets. 23 of the categories are extracted from the AMR 3.0 test set; evaluating with these metrics grants insights into the strengths and weaknesses of GPT as an AMR parser. For comparison, we include a high-performing fine-tuned BART model, AMRBart (Bai et al., 2022) (Smatch 84). Full GrAPES results are in Appendix B.3. We also ran Damonte, Cohen, and Satta (2017)'s fine-grained Smatch on all outputs (see Appendix B.1) and highlight some relevant results here.

Unsurprisingly, AMRBart outperforms GPT in nearly all categories. However, there is substantial and, we argue, principled, cross-categorical variation: overall, GPT is much worse at more complex and AMR-specific tasks. There are also effects of it not having been trained specifically on the AMR training set.

Seen vs Unseen Unlike with fine-tuned AMR parsers, GPT shows very little difference in performance on subcategories of things seen and unseen in the AMR 3.0 training set:

Category	5-shot 4o	CoT 4o	AMR Bart
Seen vs Unseen			
Rare node labels	61	57	69
Unseen node labels	**61**	56	45
Hard unseen wiki links	**33**	5	9
Seen	71	59	93
Unseen	57	48	58
Seen − Unseen as % of Seen	10%	18%	38%
AMR-specific			
PropBank	30	25	63
Multinode word meanings	14	4	84
Imperatives	4	0	66
Ellipsis	12	15	55
Special Entities	64	55	77
Average AMR-spec.	25	20	69
Average all categories	48	42	72
AMR-spec − all as % of all	49%	53%	4%

Table 2: Selection of fine-grained categories from GrAPES. *Italicised* categories are averages across multiple categories. Scores are (averages of) recall.

while AMRBart performs on average 38% worse on unseen items, our best GPT model is only 10% worse (Table 2).

Simple vs complex, AMR-specific subtasks GPT performs well on simple tasks like node labeling. For instance, fine-grained Smatch includes the F-score over the multiset of node labels, where GPT scores 67, notably higher than its overall Smatch F-score of 60. GPT even outperforms AMRBart on the GrAPES category *Hard unseen wiki links*, which are wiki links for named entities that are not templatic. Evidently, these unpredictable URLs occur in GPT's training data, and it is able to make use of them.

However, the more complex and AMR-specific the subtask, the worse GPT gets. For tasks we classified as AMR-specific (PropBank tasks, multinode word meanings (e.g. *teacher* is annotated `(person :ARG0-of teach-01)`), imperatives, ellipsis, and special entities), GPT performed 49% worse than its average GrAPES score, while AMRBart performed only 4% worse.

Two-step performance with GPT-3.5 is comparable to the best 3.5 version (5-shot, Smatch 50 and 49). Here the core roles are predicted in the 5-shot setting, and indeed the fine-grained Smatch score for *SRL* (Semantic Role Labeling, core roles) is equal at 47. Since non-core roles are predicted with CoT and CoT performs worse overall, we might expect overall poor performance for *Negation*, *NER (Named Entity Recognition)*, and *Wiki links*. In fact, performance here is inconsistent, being worse for negation (13 vs 10) and Wiki links (66 vs 59) but identical for NER (69). Also of note is that sampling indicates that step 1 outputs often contain more than just core roles. More

insights can be gained here by trying GPT-4o and performing an error analysis on sampled outputs, as we did for CoT.

5 Discussion

GPT performs poorly LLM performance is correlated with the amount of task-relevant data during pre-training (Kandpal, Deng, Roberts, Wallace, & Raffel, 2023). Even if the whole AMR 3.0 dataset slipped into the GPT training data, it only has 59,255 sentences, yielding near-zero results in the zero-shot setting. The purest version of GPT as an out-of-the-box AMR parser is therefore right out. This is in contrast to, for instance, Python programming, where Poldrack, Lu, and Beguš (2023) found that natural-language prompts for Python code were usable on the first try in 38% of cases. Parsing into Python code is arguably just as difficult a task as AMR parsing, so we might expect similar outcomes were it not for the presumably huge difference in training data quantities.

Fine-grained analysis reveals a large discrepancy between subtasks that are fairly simple and easy to predict, such as basic node labeling, and subtasks that are complex or AMR-specific, such as imperatives. Because many language phenomena have a Zipfian distribution, it is impossible to create a single, short enough example that contains every phenomenon that can – or even is likely to – arise.

A better selection of the examples in the 5-shot may help, since we only need to illustrate phenomena for one sentence at a time. However, the problem of identifying the phenomena to demonstrate, and finding the AMRs that exemplify them, is in itself a kind of parsing. A proof-of-concept experiment could use the gold AMR and measure graph similarity, but this would not be usable as an AMR parsing method.

Despite recent advancements enabling LLMs to process extended contexts (Lin et al., 2024), incorporating AMR annotation guidelines directly into the prompt (as attempted in the second step of our two-step approach) did not yield significant improvements.

Attempts to split the task using a CoT prompting strategy were also unsuccessful. This approach may require multiple CoT examples containing potentially needed non-core roles to form effective k-shot prompts. Moreover, its success is limited not only by the model's context window size but also by the difficulty of obtaining high-quality CoT examples.

Interpretability of CoT AMR parsing Although CoT uses a similar setup to one-shot, it is not an extension of the 1-shot method. The output graphs are not necessarily the same, and even when they are, there is no way to know whether the Chain-of-Thought is in any way related to how the model built the graph in the one-shot case.

An advantage of CoT is that it to a certain extent reflects how GPT derives an AMR in the CoT case, since the subgraphs in the chain of reasoning are usually in fact subgraphs of the final output. We can often easily find errors through the CoT reasoning process, which can make it easier to correct errors by hand. The difference between the subgraph and gold AMR could then be used as the loss signal for prompt

tuning. This method opens the possibility to subsequently use LLM as a generative model for data augmentation, especially for complex sentences.

6 Conclusion

We compared the capabilities of GPT models on AMR parsing under various prompting strategies. We found that GPT-3.5 and GPT-4o make poor AMR parsers, with a maximum Smatch of 60.

Two CoT prompting methods for AMR parsing (bottom-up and top-down) were introduced, as well as a two-step approach, with core and non-core roles added separately. A two-step method was also explored, splitting the task into generating core- and non-core roles. All worsen performance over 5-shot methods, but arguably add some interpretability.

A detailed analysis revealed GPT's limited ability to produce AMRs in alignment with structural AMR guidelines. These guidelines specify precise ways to annotate certain things, such as named entities, that cannot be predicted *a priori*. GPT struggles to express many linguistic phenomena within the AMR paradigm.

References

Bai, X., Chen, Y., & Zhang, Y. (2022). Graph pre-training for AMR parsing and generation. In S. Muresan, P. Nakov, & A. Villavicencio (Eds.), *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 6001–6015). Dublin, Ireland: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2022.acl-long.415 doi: 10.18653/v1/2022.acl-long.415

Blloshmi, R., Tripodi, R., & Navigli, R. (2020). XL-AMR: Enabling cross-lingual AMR parsing with transfer learning techniques. In B. Webber, T. Cohn, Y. He, & Y. Liu (Eds.), *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 2487–2500). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2020.emnlp-main.195 doi: 10.18653/v1/2020.emnlp-main.195

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... Amodei, D. (2020). Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 1877–1901). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

Cai, D., & Lam, W. (2020). AMR parsing via graph-sequence iterative inference. In D. Jurafsky, J. Chai, N. Schluter, & J. Tetreault (Eds.), *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 1290–1301). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2020.acl-main.119 doi: 10.18653/v1/2020.acl-main.119

Cai, S., & Knight, K. (2013). Smatch: an evaluation metric for semantic feature structures. In H. Schuetze, P. Fung, & M. Poesio (Eds.), *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 2: Short papers)* (pp. 748–752). Sofia, Bulgaria: Association for Computational Linguistics. Retrieved from https://aclanthology.org/P13-2131

Cao, T., Law, M. T., & Fidler, S. (2020). A theoretical analysis of the number of shots in few-shot learning. In *International conference on learning representations.* Retrieved from https://openreview.net/forum?id=HkgB2TNYPS

Choi, J. D., Bonial, C., & Palmer, M. (2010). Propbank instance annotation guidelines using a dedicated editor, jubilee. In N. Calzolari et al. (Eds.), *Proceedings of the seventh international conference on language resources and evaluation (LREC'10).* Valletta, Malta: European Language Resources Association (ELRA). Retrieved from http://www.lrec-conf.org/proceedings/lrec2010/pdf/903_Paper.pdf

Damonte, M., Cohen, S. B., & Satta, G. (2017). An incremental parser for Abstract Meaning Representation. In M. Lapata, P. Blunsom, & A. Koller (Eds.), *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: Volume 1, long papers* (pp. 536–546). Valencia, Spain: Association for Computational Linguistics. Retrieved from https://aclanthology.org/E17-1051

Dong, Q., Li, L., Dai, D., Zheng, C., Ma, J., Li, R., ... Sui, Z. (2024). A survey on in-context learning. In Y. Al-Onaizan, M. Bansal, & Y.-N. Chen (Eds.), *Proceedings of the 2024 conference on empirical methods in natural language processing* (pp. 1107–1128). Miami, Florida, USA: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2024.emnlp-main.64/ doi: 10.18653/v1/2024.emnlp-main.64

Ettinger, A., Hwang, J., Pyatkin, V., Bhagavatula, C., & Choi, Y. (2023). "you are an expert linguistic annotator": Limits of LLMs as analyzers of Abstract Meaning Representation. In H. Bouamor, J. Pino, & K. Bali (Eds.), *Findings of the association for computational linguistics: Emnlp 2023* (pp. 8250–8263). Singapore: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2023.findings-emnlp.553 doi: 10.18653/v1/2023.findings-emnlp.553

Fernandez Astudillo, R., Ballesteros, M., Naseem, T., Blodgett, A., & Florian, R. (2020). Transition-based parsing with stack-transformers. In T. Cohn, Y. He, & Y. Liu (Eds.), *Findings of the association for computational linguistics: Emnlp 2020* (pp. 1001–1007). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2020.findings-emnlp.89 doi: 10.18653/v1/2020.findings-emnlp.89

Flanigan, J., Thomson, S., Carbonell, J., Dyer, C., & Smith, N. A. (2014). A discriminative graph-based parser for the Abstract Meaning Representation. In K. Toutanova & H. Wu (Eds.), *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1426–1436). Baltimore, Maryland: Association for Computational Linguistics. Retrieved from https://aclanthology.org/P14-1134 doi: 10.3115/v1/P14-1134

Gao, T., Fisch, A., & Chen, D. (2021). Making pre-trained language models better few-shot learners. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *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)* (pp. 3816–3830). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2021.acl-long.295 doi: 10.18653/v1/2021.acl-long.295

Goodman, M. W. (2020). Penman: An open-source library and tool for AMR graphs. In A. Celikyilmaz & T.-H. Wen (Eds.), *Proceedings of the 58th annual meeting of the association for computational linguistics: System demonstrations* (pp. 312–319). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2020.acl-demos.35 doi: 10.18653/v1/2020.acl-demos.35

Groschwitz, J., Cohen, S., Donatelli, L., & Fowlie, M. (2023). AMR parsing is far from solved: GrAPES, the granular AMR parsing evaluation suite. In H. Bouamor, J. Pino, & K. Bali (Eds.), *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 10728–10752). Singapore: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2023.emnlp-main.662 doi: 10.18653/v1/2023.emnlp-main.662

Kandpal, N., Deng, H., Roberts, A., Wallace, E., & Raffel, C. (2023). Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th international conference on machine learning.* JMLR.org. Retrieved from https://dl.acm.org/doi/10.5555/3618408.3619049

Knight, K., Badarau, B., Baranescu, L., Bonial, C., Bardocz, M., Griffitt, K., ... Schneider, N. (2017). *Abstract Meaning Representation (AMR) Annotation Release 2.0.* Abacus Data Network. Retrieved from https://doi.org/10.35111/s444-np87

Knight, K., Badarau, B., Baranescu, L., Bonial, C., Bardocz, M., Griffitt, K., ... others (2021). *Abstract Meaning Representation (AMR) annotation release 3.0.* https://catalog.ldc.upenn.edu/LDC2020T02. Abacus Data Network.

Kong, A., Zhao, S., Chen, H., Li, Q., Qin, Y., Sun, R., & Zhou, X. (2023). *Better zero-shot reasoning with role-play prompting.*

Lee, Y.-S., Astudillo, R., Thanh Lam, H., Naseem, T., Florian, R., & Roukos, S. (2022). Maximum Bayes Smatch ensemble distillation for AMR parsing. In M. Carpuat, M.-C. de Marneffe, & I. V. Meza Ruiz (Eds.), *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 5379–5392). Seattle, United States: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2022.naacl-main.393 doi: 10.18653/v1/2022.naacl-main.393

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In D. Jurafsky, J. Chai, N. Schluter, & J. Tetreault (Eds.), *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 7871–7880). Online: Associa-

tion for Computational Linguistics. Retrieved from https://aclanthology.org/2020.acl-main.703/ doi: 10.18653/v1/2020.acl-main.703

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., ... Misra, V. (2022). Solving quantitative reasoning problems with language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (Vol. 35, pp. 3843–3857). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2022/file/18abbeef8cfe9203fdf9053c9c4fe191-Paper-Conference.pdf

Lin, B., Zhang, C., Peng, T., Zhao, H., Xiao, W., Sun, M., ... Lin, W. (2024). *Infinite-llm: Efficient llm service for long context with distattention and distributed kvcache.* Retrieved from https://arxiv.org/abs/2401.02669

Lindemann, M., Groschwitz, J., & Koller, A. (2020). Fast semantic parsing with well-typedness guarantees. In B. Webber, T. Cohn, Y. He, & Y. Liu (Eds.), *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 3929–3951). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2020.emnlp-main.323 doi: 10.18653/v1/2020.emnlp-main.323

Liu, J., Shen, D., Zhang, Y., Dolan, B., Carin, L., & Chen, W. (2022). What makes good in-context examples for GPT-3? In E. Agirre, M. Apidianaki, & I. Vulić (Eds.), *Proceedings of deep learning inside out (deelio 2022): The 3rd workshop on knowledge extraction and integration for deep learning architectures* (pp. 100–114). Dublin, Ireland and Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2022.deelio-1.10 doi: 10.18653/v1/2022.deelio-1.10

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, *55*(9). Retrieved from https://doi.org/10.1145/3560815 doi: 10.1145/3560815

Lyu, Q., Havaldar, S., Stein, A., Zhang, L., Rao, D., Wong, E., ... Callison-Burch, C. (2023). Faithful Chain-of-Thought reasoning. In J. C. Park et al. (Eds.), *Proceedings of the 13th international joint conference on natural language processing and the 3rd conference of the asia-pacific chapter of the association for computational linguistics (volume 1: Long papers)* (pp. 305–329). Nusa Dua, Bali: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2023.ijcnlp-main.20

Madaan, A., & Yazdanbakhsh, A. (2022). *Text and patterns: For effective chain of thought, it takes two to tango.* Retrieved from https://arxiv.org/abs/2209.07686

Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., & Zettlemoyer, L. (2022). Rethinking the role of demonstrations: What makes in-context learning work? In Y. Goldberg, Z. Kozareva, & Y. Zhang (Eds.), *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp.

11048–11064). Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2022.emnlp-main.759 doi: 10.18653/v1/2022.emnlp-main.759

Naseem, T., Shah, A., Wan, H., Florian, R., Roukos, S., & Ballesteros, M. (2019). Rewarding Smatch: Transition-based AMR parsing with reinforcement learning. In A. Korhonen, D. Traum, & L. Màrquez (Eds.), *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 4586–4592). Florence, Italy: Association for Computational Linguistics. Retrieved from https://aclanthology.org/P19-1451 doi: 10.18653/v1/P19-1451

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., ... Zoph, B. (2024). *Gpt-4 technical report.* Retrieved from https://arxiv.org/abs/2303.08774

Peng, X., Gildea, D., & Satta, G. (2018). AMR parsing with cache transition systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1). Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/11922 doi: 10.1609/aaai.v32i1.11922

Poldrack, R. A., Lu, T., & Beguš, G. (2023). *Ai-assisted coding: Experiments with GPT-4.* Retrieved from https://arxiv.org/abs/2304.13187

Reynolds, L., & McDonell, K. (2021). Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended abstracts of the 2021 chi conference on human factors in computing systems.* New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3411763.3451760 doi: 10.1145/3411763.3451760

Saparov, A., & He, H. (2023). Language models are greedy reasoners: A systematic formal analysis of Chain-of-Thought. In *The eleventh international conference on learning representations.* Retrieved from https://openreview.net/forum?id=qFVVBzXxR2V

van Noord, R., & Bos, J. (2017). The meaning factory at SemEval-2017 task 9: Producing AMRs with neural semantic parsing. In S. Bethard, M. Carpuat, M. Apidianaki, S. M. Mohammad, D. Cer, & D. Jurgens (Eds.), *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)* (pp. 929–933). Vancouver, Canada: Association for Computational Linguistics. Retrieved from https://aclanthology.org/S17-2160 doi: 10.18653/v1/S17-2160

Vasylenko, P., Huguet Cabot, P. L., Martínez Lorenzo, A. C., & Navigli, R. (2023). Incorporating graph information in transformer-based AMR parsing. In A. Rogers, J. Boyd-Graber, & N. Okazaki (Eds.), *Findings of the association for computational linguistics: Acl 2023* (pp. 1995–2011). Toronto, Canada: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2023.findings-acl.125 doi: 10.18653/v1/2023.findings-acl.125

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 5776–5788). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/

paper/2020/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., ... Fedus, W. (2022). Emergent abilities of large language models. *Transactions on Machine Learning Research*. Retrieved from https://openreview.net/forum?id=yzkSU5zdwD (Survey Certification)

Wei, J., Wang, X., Schuurmans, D., Bosma, M., ichter, b., Xia, F., ... Zhou, D. (2022). Chain-of-Thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (Vol. 35, pp. 24824–24837). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf

Weng, Y., Zhu, M., Xia, F., Li, B., He, S., Liu, S., ... Zhao, J. (2023). Large language models are better reasoners with self-verification. In H. Bouamor, J. Pino, & K. Bali (Eds.), *Findings of the association for computational linguistics: Emnlp 2023* (pp. 2550–2575). Singapore: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2023.findings-emnlp.167 doi: 10.18653/v1/2023.findings-emnlp.167

Yang, M. S., Schuurmans, D., Abbeel, P., & Nachum, O. (2022). Chain of Thought imitation with procedure cloning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (Vol. 35, pp. 36366–36381). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2022/file/ebdb990471f653dffb425eff03c7c980-Paper-Conference.pdf

Zhang, S., Ma, X., Duh, K., & Van Durme, B. (2019). Broad-coverage semantic parsing as transduction. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 3786–3798). Hong Kong, China: Association for Computational Linguistics. Retrieved from https://aclanthology.org/D19-1392 doi: 10.18653/v1/D19-1392

Zhao, Z., Wallace, E., Feng, S., Klein, D., & Singh, S. (2021). Calibrate before use: Improving few-shot performance of language models. In M. Meila & T. Zhang (Eds.), *Proceedings of the 38th international conference on machine learning* (Vol. 139, pp. 12697–12706). PMLR. Retrieved from https://proceedings.mlr.press/v139/zhao21c.html

Zhou, J., Naseem, T., Fernandez Astudillo, R., & Florian, R. (2021). AMR parsing with action-pointer transformer. In K. Toutanova et al. (Eds.), *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 5585–5598). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2021.naacl-main.443 doi: 10.18653/v1/2021.naacl-main.443

A Appendix: Prompt Design Detail

A.1 0-shot Prompt

```
You are a computational linguist. Let's do some semantic parsing.
Q: Please give me an Abstract Meaning Representation (AMR) of '[Target Sentence]'.
A:
```

A.2 Few-shot Prompt

A.2.1 1-shot Prompt

```
You are a computational linguist. Let's do some semantic parsing.
Q: Please give me an Abstract Meaning Representation (AMR) of "The poor kid
didn't receive the gift and the postcard that Dorothy Gale sent him on May 25th.".
A:
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard)
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person
                        :wiki "Dorothy_Gale"
                        :name (n / name
                              :op1 "Dorothy"
                              :op2 "Gale"))
                  :ARG1 k
                  :time (d / date-entity
                        :month 5
                        :date 25)))
      :polarity -)
Q: Please give me an Abstract Meaning Representation (AMR) of '[Target Sentence]'.
A:
```

A.2.2 Example 5-shot Prompt

```
You are a computational linguist. Let's do some semantic parsing.
Q: Please give me an Abstract Meaning Representation (AMR) of 'However, when it
comes to the soul and the physical being, in the Church's eyes, the soul trumps
the former.'.
A:
(c2 / contrast-01
      :ARG2 (t / trump-01
            :ARG0 (s2 / soul)
            :ARG1 b
            :ARG1-of (v / view-02
                  :ARG0 (o / organization
                        :wiki "Catholic_Church"
```

```
                        :name (n / name
                            :op1 "Church")))
            :condition (c3 / come-12
                  :ARG1 (a / and
                        :op1 (s / soul)
                        :op2 (b / be-02
                              :mod
                              (p / physical))))))

Q: Please give me an Abstract Meaning Representation (AMR) of 'Thanks for the
article and the background - I read both.'.
A:
(m / multi-sentence
      :snt1 (t / thank-01
            :ARG2 (a / and
                  :op1 (a2 / article)
                  :op2 (b / background)))
      :snt2 (r / read-01
            :ARG0 (i / i)
            :ARG1 (b2 / both)))

Q: Please give me an Abstract Meaning Representation (AMR) of 'International;
weapons; proliferation; dissent; politics.'.
A:
(a / and
      :op1 (i / international)
      :op2 (w / weapon)
      :op3 (p / proliferation)
      :op4 (d / dissent-01)
      :op5 (p2 / politics))

Q: Please give me an Abstract Meaning Representation (AMR) of 'Speakers addressed
opium production and trafficking in Afghanistan.'.
A:
(a / address-01
      :ARG0 (p / person
            :ARG0-of (s / speak-01))
      :ARG1 (a2 / and
            :op1 (p2 / produce-01
                  :ARG1 (o / opium))
            :op2 (t / traffic-01)
            :location (c / country
                    :wiki "Afghanistan"
                    :name (n / name :op1
                            "Afghanistan"))))

Q: Please give me an Abstract Meaning Representation (AMR) of 'I'm sorry for the
swearing, i'm not one to usually swear, but the CIA et al are a joke.'.
A:
(a2 / and
      :op1 (s / sorry-01
            :ARG1 (i / i)
```

```
            :ARG2 (s2 / swear-02
                  :ARG0 i))
      :op2 (s3 / swear-02 :polarity -
            :ARG0 i
            :mod (u / usual)
            :concession-of (j / joke
                  :domain (a / and
                        :op1 (g /
                        government-organization
                        :wiki "Central
                        _Intelligence
                        _Agency"
                            :name (n / name
                            :op1 "CIA"))
                            :op2 (o / other)))))

Q: Please give me an Abstract Meaning Representation (AMR) of 'well, where did
it go in the first place?'.
A:
(g / go-01
      :ARG1 (i / it)
      :ARG4 (a / amr-unknown)
      :prep-in (p / place
            :ord (o / ordinal-entity :value 1)))

Q: Please give me an Abstract Meaning Representation (AMR) of '[Target Sentence]'.
A:
```

A.3 Top-down CoT Prompt

```
You are a computational linguist. Let's do some semantic parsing.
Q: Please give me an Abstract Meaning Representation (AMR) of "The poor kid
didn't receive the gift and the postcard that Dorothy Gale sent him on May 25th.".
A:
1. "didn't receive": This is the main verb phrase of the sentence, indicating
that "the poor kid" did not receive something. "the gift and the postcard"
refers to the direct object of the verb "receive", specifying what the poor kid
didn't receive.
(r / receive-01
      :ARG0 (k / kid)
      :ARG1 (a / and)
      :polarity -)
2. "the poor kid" indicates that "kid" is poor.
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and)
      :polarity -)
3. "the gift and the postcard" are the two objects that the poor kid didn't

receive.
(r / receive-01
```

```
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard))
      :polarity -)

4. "that Dorothy Gale sent him on May 25th": This relative clause modifies "the
gift and the postcard" which are sent by a person ("Dorothy Gale") to the kid.
The kid is the recipient of "the gift and the postcard". Hence, there is a
reentrancy at node k. "sent him on May 25th": The action "sent" happened on May
25th, which is a date-entity. The ":month" is 5 and the ":date" is 25.
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard)
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person)
                  :ARG1 k
                  :time (d / date-entity
                        :month 5
                        :date 25)))
      :polarity -)
5.  "Dorothy Gale sent" indicates the person who sent "the gift and the postcard"
is named "Dorothy Gale".
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard)
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person
                        :wiki "Dorothy_Gale"
                        :name (n / name
                              :op1 "Dorothy"
                              :op2 "Gale"))
                  :ARG1 k
                  :time (d / date-entity
                        :month 5
                        :date 25)))
      :polarity -)

The AMR is:
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard)
```

```
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person
                        :wiki "Dorothy_Gale"
                        :name (n / name
                              :op1 "Dorothy"
                              :op2 "Gale"))
                  :ARG1 k
                  :time (d / date-entity
                        :month 5
                        :date 25)))
      :polarity -)

Q: Please give me an Abstract Meaning Representation (AMR) of '[Target Sentence]'.
A:
```

A.4 Bottom-up CoT Prompt

```
You are a computational linguist. Let's do some semantic parsing.
Q: Please give me an Abstract Meaning Representation (AMR) of "The poor kid
didn't receive the gift and the postcard that Dorothy Gale sent him on May 25th.".
A:
1.  This sentence has the following components: "The poor kid": indicates that
"kid" is poor.
(k / kid
      :mod (p / poor))
"the gift and the postcard": "the gift" and "the postcard" are in an "and"
relationship.
(a / and
      :op1 (g/ gift)
      :op2 (p2 / postcard))
"Dorothy Gale": "Dorothy Gale" is the "name" of a person.
(p3 / person
      :wiki "Dorothy_Gale"
      :name (n / name
            :op1 "Dorothy"
            :op2 "Gale"))
"him": refers to the "The poor kid".
k
"on May 25th": is a date-entity. The ":month" is 5 and the ":date" is 25.
(date-entity
      :month 5
      :day 25)
2. "didn't receive": This is the main verb phrase of the sentence, indicating
that "the poor kid" did not receive something. "the gift and the postcard" is
the direct object of the verb "receive" specifying what the poor kid didn't
receive.
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
```

```
            :op2 (p2 / postcard))
      :polarity -)
3. "that Dorothy Gale sent him on May 25th": The action "sent" is performed by a
person named "Dorothy Gale" to the kid, and it happened on May 25th. The kid is
the recipient of "the gift and the postcard". Hence, there is a reentrancy at
node k.
(s / send-01
      :ARG0 (p3 / person
            :wiki "Dorothy_Gale"
            :name (n / name
                :op1 "Dorothy"
                :op2 "Gale"))
      :ARG1 k
      :time (d / date-entity
            :month 5
            :date 25))
4. "the gift and the postcard that Dorothy Gale sent him on May 25th": "the gift
and the postcard" is the object of "sent". This is a relative clause, so we
make "sent" an "ARG1-of" belonging to "and".
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard)
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person
                        :wiki "Dorothy_Gale"
                        :name (n / name
                            :op1 "Dorothy"
                            :op2 "Gale"))
                  :ARG1 k
                  :time (d / date-entity
                        :month 5
                        :date 25)))
      :polarity -)
The AMR is:
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard)
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person
                        :wiki "Dorothy_Gale"
                        :name (n / name
                              :op1 "Dorothy"
                              :op2 "Gale"))
                  :ARG1 k
                  :time (d / date-entity
                        :month 5
```

```
                              :date 25)))
      :polarity -)
Q: Please give me an Abstract Meaning Representation (AMR) of '[Target Sentence]'.
A:
```

A.5 Two-step Version: Prompt Design Detail

Since there are hundreds of non-core roles of AMR and prompts have token limitations, it is impossible to provide sufficient numbers of examples in one single sentence or in one CoT prompt.

To address this problem, we break the whole AMR generation process into two parts, firstly generate core roles (e.g. the ARGs) and then add non-core roles in a separate step.

Step 1 is to generate core roles with 5-shot prompting, where the prompts only provide AMR with core roles. There are two different AMR styles of Step 1, one is simply pruning all non-core roles in the examples, which might break an AMR graph into several sub-graphs (notated as "multi-graph"). Another is only pruning the non-core roles leaf nodes, which might be some remnants of non-core roles in the AMR graph, but the AMR graph will remain as a single graph (notated as "one-graph"). A "one-graph" style example is shown in A.5.1.

Step 2 is a CoT prompt with an example and a guideline that includes several non-core roles, an example is shown in A.5.2. The results are shown in Table 7 and Table 9.

A.5.1 Step 1: Example 5-shot Prompt for generating core roles

```
You are a computational linguist. Let's do some semantic parsing.
Q: Please give me an Abstract Meaning Representation (AMR) of 'However, when it
comes to the soul and the physical being, in the Church's eyes, the soul trumps
the former.'.
A:
(c2 / contrast-01
      :ARG2 (t / trump-01
            :ARG0 (s2 / soul)
            :ARG1 b
            :ARG1-of (v / view-02
                  :ARG0 (o / organization))
            :condition (c3 / come-12
                  :ARG1 (a / and))))

Q: Please give me an Abstract Meaning Representation (AMR) of 'Thanks for the
article and the background - I read both.'.
A:
(m / multi-sentence
      :snt1 (t / thank-01
            :ARG2 (a / and))
      :snt2 (r / read-01
```

```
            :ARG0 (i / i)
            :ARG1 (b2 / both)))

Q: Please give me an Abstract Meaning Representation (AMR) of 'International;
weapons; proliferation; dissent; politics.'.
A:
(a / and)

Q: Please give me an Abstract Meaning Representation (AMR) of 'Speakers addressed
opium production and trafficking in Afghanistan.'.
A:
(a / address-01
      :ARG0 (p / person
            :ARG0-of (s / speak-01))
      :ARG1 (a2 / and
            :op1 (p2 / produce-01
                  :ARG1 (o / opium))))

Q: Please give me an Abstract Meaning Representation (AMR) of 'I'm sorry for the
swearing, i'm not one to usually swear, but the CIA et al are a joke.'.
A:
(a2 / and
      :op1 (s / sorry-01
            :ARG1 (i / i)
            :ARG2 (s2 / swear-02
                  :ARG0 i))
      :op2 (s3 / swear-02 :polarity -
            :ARG0 i))

Q: Please give me an Abstract Meaning Representation (AMR) of 'well, where did
it go in the first place?'.
A:
(g / go-01
      :ARG1 (i / it)
      :ARG4 (a / amr-unknown))

Q: Please give me an Abstract Meaning Representation (AMR) of '[Target Sentence]'.
A:
```

A.5.2 Step 2: CoT with guildline

```
You are a computational linguist. Let's do some semantic parsing.
non-core roles list:
:accompanier, :age, :beneficiary, :concession, :condition, :consist-of, :degree,
:destination, :direction, :domain, :duration, :example, :extent, :frequency,
:instrument, :li, :location, :manner, :medium, :mod, :mode, :name, :ord, :part,
:path, :polarity, :polite, :poss, :purpose, :quant,
:range, :scale, :source, :subevent, :time,
:topic,
:unit, :value, :wiki
date-entity:
```

```
:calendar, :century, :day, :dayperiod, :decade, :era, :month, :quarter, :season,
:timezone, :weekday, :year, :year2
conjunctions:
:op1, :op2, :op3, :op4, ...
other components:
:country

Q: Here is the sentence and its half-completed AMR graph.
Sentence:
'The poor kid didn't receive the gift and the postcard that Dorothy Gale sent him
on May 25th.'

Half-completed AMR graph:
(r / receive-01
      :ARG0 (k / kid)
      :ARG1 (a / and
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person)
                  :ARG1 k)))

Please give me an Abstract Meaning Representation (AMR) of the sentence according
to its half-completed AMR graph, and aforementioned non-core roles list.
A:
"(r / receive-01)" requires a ":polarity -" because the receiving "didn't" happen.
(r / receive-01
      :polarity -)

For ":ARG0 (k / kid)", because the "kid" is "poor", it is a modifier. That is the
":mod" in the non-core roles list, so the ":mod" is "(p / poor)".
:ARG0 (k / kid
      :mod (p / poor))

"the gift and the postcard" is a conjunction, so ":ARG1 (a / and)" has "op1" and
"op2" from the list; the ":op1" is "(g/ gift)" and the ":op2" is "(p2 / postcard)"
:ARG1 (a / and
      :op1 (g/ gift)
      :op2 (p2 / postcard))

For ":ARG0-of (s / send-01)", "sent him on May 25th", indicates that the sending
happened on May 25th, which is a time. We use ":time" in the non-core roles list,
and it is a "date-entity"; the ":month" is 5 and the ":day" is 25.
:ARG1-of (s / send-01
      :ARG0 (p3 / person)
      :ARG1 k
      :time (d / date-entity
            :month 5
            :date 25))

For ":ARG0 (p3 / person)",  "Dorothy Gale", is a name, which is ":name" in the
non-core roles list, and also ":wiki" in the non-core roles list, so ":wiki" is
"Dorothy_Gale". For the ":name", the two parts of this name are introduced with
"op1" and "ops2". ":op1" is "Dorothy" and the ":op2" is "Gale".
```

```
:ARG0 (p3 / person
      :wiki "Dorothy Gale"
      :name (n / name
            :op1 "Dorothy"
            :op2 "Gale"))

The AMR is:
(r / receive-01
      :ARG0 (k / kid
            :mod (p / poor))
      :ARG1 (a / and
            :op1 (g/ gift)
            :op2 (p2 / postcard)
            :ARG1-of (s / send-01
                  :ARG0 (p3 / person
                        :wiki "Dorothy_Gale"
                        :name (n / name
                              :op1 "Dorothy"
                              :op2 "Gale"))
                  :ARG1 k
                  :time (d / date-entity
                        :month 5
                        :date 25)))
      :polarity -)

Q: Here is the sentence and its half-completed AMR graph.
Sentence:
'[Target_Sentence]'

Half-completed AMR graph:
[Target_Graph]

Please give me an Abstract Meaning Representation (AMR) of the sentence according
to its half-completed AMR graph, and aforementioned non-core roles list.
A:
```

B Appendix: Full Results

B.1 Fine-grained Smatch Results

Smatch sub-metrics	Definition
Unlabeled (Unlab.)	Smatch score after pruning the edge labels.
NoWSD	Smatch score which ignores Propbank senses.
Concepts (Con.)	F-score on the concept identification task.
Named Entity Recognition (NER.)	F-score on the named entity recognition.
Negations (Neg.)	F-score on the negation detection.
Wikification (Wiki.)	F-score on the wikification.
Semantic Role Labeling (SRL.)	Smatch score computed on :ARG-i roles only.
Reentrancy (Reen.)	Smatch score on reentrant edges only.

Table 3: Fine-grained Smatch definition (Damonte et al., 2017).[5]

		Model	Smatch	Unlab.	NoWSD	Con.	NER.	Neg.	Wiki.	Reen.	SRL.
Baseline	0-shot	turbo-instruct	3	4	3	4	3	0	4	2	4
		davinci	6	7	6	7	4	1	4	2	8
	1-shot	turbo-instruct	28	34	28	33	27	3	19	17	29
		davinci	32	39	33	36	34	9	40	21	33
	5-shot random	turbo-instruct	34	41	35	**43**	35	3	38	19	**35**
		davinci	**37**	**44**	**38**	**43**	40	8	**48**	21	**35**
	5-shot similarity	turbo-instruct	33	39	34	39	41	7	39	20	32
		davinci	**37**	43	**38**	42	**51**	10	**48**	**23**	**35**
CoT approach	CoT top-down	turbo-instruct	14	17	14	15	10	5	7	6	14
		davinci	27	32	27	30	12	13	14	16	30
	CoT bottom-up	turbo-instruct	12	15	12	13	15	7	11	4	12
		davinci	24	29	25	28	19	**14**	16	14	24

Table 4: Fine-grained Smatch result of baseline and CoT approach (AMR 2.0, raw output).

		Model	Smatch	Unlab.	NoWSD	Con.	NER.	Neg.	Wiki.	Reen.	SRL.
Baseline	0-shot	turbo-instruct	5	7	6	7	5	0	1	3	7
		davinci	14	18	14	16	10	1	8	8	18
	1-shot	turbo-instruct	39	48	40	47	39	6	28	27	40
		davinci	41	50	42	47	42	11	51	29	42
	5-shot random	turbo-instruct	42	50	43	52	42	4	47	25	43
		davinci	44	52	45	51	48	9	57	26	42
	5-shot similarity	turbo-instruct	44	52	46	53	55	10	54	30	43
		davinci	**50**	**58**	**51**	**57**	**69**	13	**66**	**34**	**47**
CoT approach	CoT top-down	turbo-instruct	34	44	34	37	23	11	17	22	36
		davinci	36	45	37	41	17	14	22	25	40
	CoT bottom-up	turbo-instruct	30	37	30	34	30	13	24	15	31
		davinci	33	41	34	39	26	**19**	24	22	34

Table 5: Fine-grained Smatch result of baseline and CoT approach (AMR 2.0, post-processed).

[5] https://github.com/mdtux89/amr-evaluation

Method	Evaluation object	Smatch	Unlab.	NoWSD	Con.	NER.	Neg.	Wiki.	Reen.	SRL.
5-shot similarity	well-formed AMR only	61	68	63	69	75	33	71	42	57
	raw output	53	59	54	59	66	28	62	35	49
	post-processed	60	67	61	67	73	32	69	41	56
CoT top-down	well-formed AMR only	58	66	59	65	72	37	29	38	55
	raw output	43	48	44	47	55	27	19	26	39
	post-processed	55	63	56	62	70	34	25	36	52

Table 6: Fine-grained Smatch result on the GPT-4o model (AMR 3.0).

B.1.1 Two-step Version Result

Step 1 style	Model name	Smatch	Unlab.	NoWSD	Con.	NER.	Neg.	Wiki.	Reen.	SRL.
one_graph	turbo-instruct	12	14	12	14	16	3	13	7	13
	davinci	**24**	**29**	**25**	**28**	**30**	**8**	**25**	**17**	**26**
multi_graphs	turbo-instruct	5	5	5	5	4	3	1	2	5
	davinci	11	13	11	13	10	4	5	5	12

Table 7: Fine-grained Smatch result of two-step version (AMR 2.0, raw output).

Step 1 style	Model name	Smatch	Unlab.	NoWSD	Con.	NER.	Neg.	Wiki.	Reen.	SRL.
one_graph	turbo-instruct	41	53	42	51	61	13	51	27	**47**
	davinci	**49**	**59**	**50**	**58**	**69**	**10**	**59**	**39**	**47**
multi_graphs	turbo-instruct	32	44	33	44	35	12	29	21	41
	davinci	39	50	40	49	50	14	35	31	41

Table 8: Fine-grained Smatch result of two-step version (AMR 2.0, post-processed).

B.2 Fine-grained result comparison (AMR 2.0)

	Method	Model	Smatch	Unlab.	NoWSD	Con.	NER.	Neg.	Wiki.	Reen.	SRL.
Baseline	0-shot	davinci	6	7	6	7	4	1	4	2	8
	1-shot	davinci	32	39	33	36	34	9	40	21	33
	5-shot (similarity)	davinci	**37**	**43**	**38**	**42**	**51**	10	**48**	**23**	**35**
CoT	CoT top-down	davinci	27	32	27	30	12	**13**	14	16	30
Two-step approach	one_graph	davinci	24	29	25	28	30	8	25	17	26

Table 9: Fine-grained Smatch result comparison among baseline, CoT, and two-step version (AMR 2.0, raw output).

	Method	Model	Smatch	Unlab.	NoWSD	Con.	NER.	Neg.	Wiki.	Reen.	SRL.
Baseline	0-shot	davinci	14	18	14	16	10	1	8	8	18
	1-shot	davinci	41	50	42	47	42	11	51	29	42
	5-shot (similarity)	davinci	**50**	58	**51**	57	**69**	13	**66**	34	**47**
CoT	CoT top-down	davinci	36	45	37	41	17	**14**	22	25	40
Two-step approach	one_graph	davinci	49	**59**	50	**58**	**69**	10	59	**39**	**47**

Table 10: Fine-grained Smatch result comparison among baseline, CoT, and two-step version (AMR 2.0, post-processed).

B.3 GrAPES results

Set ID	Dataset	Metric	5-shot	CoT	AMRBart	#
1	**Pragmatic reentrancies**					
	Pragmatic coreference (testset)	Edge recall	08 [03, 22]	06 [02, 18]	39 [25, 55]	36
		Prerequisites	19 [10, 35]	22 [12, 38]	61 [45, 75]	36
2	**Unambiguous reentrancies**					
	Syntactic (gap) reentrancies	Edge recall	15 [07, 28]	27 [16, 42]	49 [34, 64]	41
		Prerequisites	54 [39, 68]	39 [26, 54]	68 [53, 80]	41
	Unambiguous coreference	Edge recall	39 [24, 56]	23 [11, 40]	65 [47, 79]	31
		Prerequisites	61 [44, 76]	52 [35, 68]	77 [60, 89]	31
4	**Rare and unseen words**					
	Rare node labels	Label recall	61 [57, 64]	57 [53, 61]	69 [66, 73]	676
	Unseen node labels	Label recall	61 [52, 69]	56 [47, 65]	45 [37, 54]	117
	Rare predicate senses (excl. -01)	Label recall	21 [13, 34]	18 [10, 30]	45 [32, 58]	56
		Prerequisites	82 [70, 90]	73 [60, 83]	91 [81, 96]	56
	Rare edge labels (ARG2+)	Edge recall	15 [07, 29]	12 [05, 26]	35 [22, 50]	40
		Prerequisites	35 [22, 50]	35 [22, 50]	72 [57, 84]	40
5	**Special entities**					
	Seen names	Recall	69 [67, 71]	71 [69, 73]	94 [93, 95]	1788
	Unseen names	Recall	70 [67, 73]	67 [64, 70]	76 [73, 79]	910
	Seen dates	Recall	68 [62, 73]	66 [59, 71]	94 [90, 96]	233
	Unseen dates	Recall	51 [45, 58]	56 [49, 63]	86 [81, 90]	204
	Other seen entities	Recall	88 [83, 91]	79 [73, 84]	97 [94, 99]	237
	Other unseen entities	Recall	88 [81, 93]	70 [61, 78]	78 [69, 85]	109
6	**Entity classification and linking**					
	Types of seen named entities	Recall	59 [57, 62]	61 [58, 63]	92 [90, 93]	1628
		Prerequisites	67 [64, 69]	69 [67, 71]	94 [93, 95]	1628
	Types of unseen named entities	Recall	39 [35, 43]	36 [32, 40]	51 [47, 55]	659
		Prerequisites	60 [56, 64]	57 [53, 61]	70 [66, 73]	659
	Seen and/or easy wiki links	Recall	73 [71, 75]	19 [17, 21]	87 [85, 88]	2064
	Hard unseen wiki links	Recall	33 [28, 39]	05 [03, 08]	09 [06, 13]	277
7	**Lexical disambiguation**					
	Frequent predicate senses (incl. -01)	Label recall	46 [43, 48]	39 [36, 41]	86 [84, 88]	1654
		Prerequisites	78 [76, 80]	73 [70, 75]	94 [93, 95]	1654
	Passives	Edge recall	47 [37, 58]	28 [19, 38]	76 [66, 84]	83
		Prerequisites	57 [46, 67]	39 [29, 49]	80 [70, 87]	83
	Unaccusatives	Edge recall	21 [12, 34]	27 [17, 41]	71 [57, 82]	48
		Prerequisites	52 [38, 66]	48 [34, 62]	79 [66, 88]	48
9	**Non-trivial word-to-node relations**					
	Ellipsis	Recall	12 [05, 27]	15 [07, 31]	55 [38, 70]	33
		Prerequisites	58 [41, 73]	45 [30, 62]	94 [80, 98]	33
	Multinode word meanings	Recall	14 [07, 26]	04 [01, 13]	84 [71, 92]	50
	Imperatives	Recall	04 [01, 11]	00 [00, 05]	66 [55, 75]	76
		Prerequisite	66 [55, 75]	59 [48, 70]	89 [81, 95]	76

Table 11: Results on all GrAPES categories extracted from the AMR 3.0 test set. Grey numbers in square brackets are 95%-Wilson confidence intervals. Our 5-shot and CoT results are for GPT-4o with similarity sampling and top-down methods respectively. AMRBart (Bai et al., 2022) is a competitive fine-tuned BART model.

One GrAPES metric we categorize as AMR-specific deserves special mention: Special Entities, which are things like dates, URLs, and names. GPT does much better here than on other AMR-specific subtasks at 64 (the average of the remaining drops to 15,

yielding an average difference from the average GrAPES score of 69%). We suspect GPT succeeds here because this task is very templatic.

C Appendix: CoT Samples

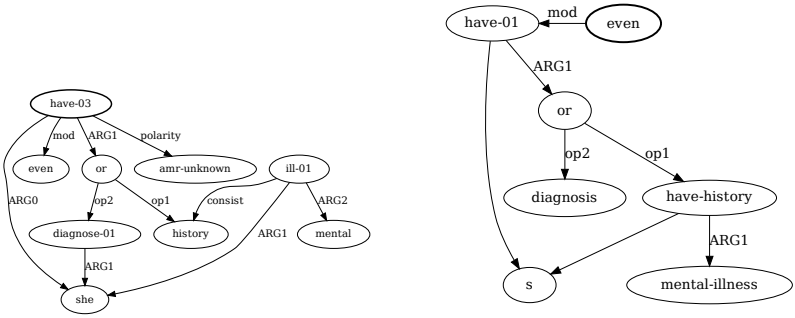
C.1 Error analysis: Chain-of-Thought

Chain-of-Thought prompting can be taken to lend some interpretability to LLMs. To that end, from the AMR 2.0 test set, we randomly sampled 6 sentences and did an error analysis of the top-down and bottom-up CoT outputs of GPT-3.5. No output was without mistakes; the 96 mistakes noted fall into 10 broad categories (Table 12). To get an idea of what this can look like, consider the bottom-up parse of example (1) in Figure 3. The gold and predicted AMRs are visualized in Figure 4. Steps in bottom-up parses were largely unnumbered, so for ease of exposition, steps are annotated (e.g. [a]).

(1) *Does she even have a history of mental illness or a diagnosis?*

```
[a] 1. This sentence has the following components: "Does she": indicates that
"she" is the subject of the sentence.
    (s / she)
[b] "even": indicates that the sentence is asking a question.
    (e / even)
[c] "have": is the main verb phrase of the sentence, indicating that "she" has
something.
    (h / have-01
          :ARG0 s)
[d] "a history of mental illness": indicates that "mental illness" is the object
of the verb "have".
    (h2 / have-history
          :ARG0 s
          :ARG1 (m / mental-illness))
[e] "or a diagnosis": indicates that "diagnosis" is an alternative to "mental
illness".
    (o / or
          :op1 h2
          :op2 (d / diagnosis))
[f] 2. The AMR is:
    (e / even
          :mod (h / have-01
                :ARG0 s
                :ARG1 (o / or
                      :op1 (h2 / have-history
                         :ARG0 s
                         :ARG1
                         (m / mental-illness))
                      :op2 (d / diagnosis))))
```

Figure 3: CoT bottom-up output for (1)



(a) Gold AMR **(b)** Predicted AMR

Figure 4: AMRs for example (1)

Overall, we can see that the subgraphs generated are not unreasonable, usually make it into the final AMR, and relate at least somewhat to the CoT text. This was true of every item sampled. Step [a] is in fact good, with *she* correctly identified and a correct AMR generated for it. All samples started with the correct `ARG0` in AMR.

Step [b], however, is very bad. Not only does *even* have nothing to do with this being a question, but no interrogative marker is added (a Missing Component error). Thus Step [b] has two CoT errors: it does not make sense, and the text does not match the subgraph. Step [c] is good, only making a PropBank sense error (`have-01` *vs* `-03`). Note as well that the `ARG0` is correctly made node `s`. In Penman notation, only one instance of a node can be labeled as in `(s / she)`; any other mentions of it should just be the node name, here `s`. No outputs we sampled had double-labeled nodes.

In step [d] we see two common errors. First, notice the node labeled `mental-illness`, which should be a complex subgraph with `ill-01` and `mental`. We found this kind of "simplification" error 12 times. Node `h2`, labeled `have-history`, is another simplification error, and it is also a CoT error, because the CoT text refers to the verb *have*, but instead of using the existing node `h`, it adds a new node `h2`. Adding a new version of an existing node happened 4 times in the sample, and never was an existing node label updated to reflect the new step's analysis.

In step [e], *diagnosis* should be annotated as a PropBank predicate `diagnose-01`. "Predicate" errors like this occurred 6 times in the sample. Perhaps node `h2` should be thought of as a `history` node; if so, the subgraph dominated by `or` is correct; however, the CoT text here does not match: it should mean `op1` is node `m`, not node `h2`, as it refers to *mental illness*.

In step [f], the subgraphs are put together. Note first that the `ARG0 (s / she)` is missing. Such "Missing earlier component" errors occur 3 times in the sample.

Conversely, the `h2` daughter of `(o / or)` has correctly substituted in the full subgraph dominated by `h2`.

The `even` node has incorrectly been made the root. Its edge to `h` is close: the correct graph has a `mod` edge from `even` to `have-03`.

Category	BU	TD	Sum
CoT	10	9	19
PropBank sense	4	3	7
Simplification	8	4	12
Predicate	3	3	6
Missing component	8	7	15
Missing earlier component	3	0	3
Named Entity	4	3	7
Duplication	2	2	4
Whole AMR in step 1	0	2	2
Other	2	18	23
Total	46	50	96
Steps	31	21	52

Table 12: CoT Error classification over 6 sentences (GPT-3.5; BU = bottom up, TD = top down).

C.2 Polarity Change Analysis

```
1. This sentence has the following components: "they": refers to a group of
people.
(p / person
      :quant 2)
"not realise": This is the main verb phrase of the sentence, indicating that the
group of people do not realize something.
(r / realize-01
      :ARG0 (p / person
            :quant 2)
      :polarity -)
"that": introduces a subordinate clause.
2. "you don't get piles of change": This is the subordinate clause, indicating
that "you" do not receive a large amount of change.
(r2 / receive-01
```

```
      :ARG0 you
      :ARG1 (c / change
            :quant (p2 / pile)))
"from the bank": specifies where the change is not received from.
(f / from
      :ARG0 c
      :ARG1 (b / bank))
3. "Do they not realise that you don't get piles of change from the bank.": The
main verb phrase is "realise", and the subordinate clause is the object of
"realise". The subordinate clause is also negated, so there is a polarity change
at the top level.
(r / realize-01
      :ARG0 (p / person
            :quant 2)
      :ARG1 (r2 / receive-01
            :ARG0 you
            :ARG1 (c / change
                  :quant (p2 / pile))
            :ARG2 (f / from
                  :ARG0 c
                  :ARG1 (b / bank)))
      :polarity +)
The AMR is:
(r / realize-01
      :ARG0 (p / person
            :quant 2)
      :ARG1 (r2 / receive-01
            :ARG0 you
            :ARG1 (c / change
                  :quant (p2 / pile))
            :ARG2 (f / from
                  :ARG0 c
                  :ARG1 (b / bank)))
      :polarity +)

Golden:
(r / realize-01 :polarity - :mode interrogative
      :ARG0 (t / they)
      :ARG1 (g / get-01 :polarity -
            :ARG0 (y / you)
            :ARG1 (p / pile
                  :mod (c / change))
            :ARG2 (b / bank)))
```

Figure 5: CoT sampling: polarity change in ex. (2)

Figure C.2 is the reasoning steps sampled from a top-down CoT generated by turbo-instruct model. The target sentence is in example (2).

(2) *Do they not realise that you don't get piles of change from the bank.*

In the earlier step, GPT added a `(:polarity -)`, because there is a *"not"* in the text. But in the later step, GPT stated that there was another negation, so it needed to switch the polarity. The explanation seems sort of reasonable. This polarity switching is not something AMR does, and was not in the example CoT; GPT 'learned' this by itself. However, here GPT is wrong anyway: the second *not* negates *get*, so there should instead be a second `(:polarity -)` in the AMR.

Correspondence

Yanming Li

Inria Saclay, INSA CVL, Paris-Saclay University
PETSCRAFT
Saclay, France
yanming.li@inria.fr

Meaghan Fowlie

Utrecht University
Department of Languages, Literature and Communication
Utrecht, the Netherlands
m.fowlie@uu.nl