Analyzing the Role of Semantic Representations in the Era of Large Language Models

Anonymous ACL submission

Abstract

Traditionally, natural language processing (NLP) models often use a rich set of features created by linguistic expertise. A typical example is the semantic representation, which turns a text into a structured graph representing the relations among the concepts and entities mentioned in the text. However, in the era of large language models (LLMs), more and more tasks are turned into a generic, end-to-end sequence generation problem. In this paper, we investigate the question – are the linguistically-grounded semantic representations of text still needed for NLP tasks? Specifically, we explore five diverse NLP tasks and provide a comprehensive analysis of cases where semantic representations are needed or not for task performance. We incorporate extensive text feature analyses to understand both cases and conduct case studies to inspect the in-depth reasons behind them. Our study provides some insights and suggestions for future NLP researchers to look at the role of the traditional semantic representations in the era of LLMs.[1]

1 Introduction

Establishing a richer structure of language beyond the raw text sequence is a fundamental pursuit in linguistics. Semantic representations, such as the abstract meaning representation (AMR) (Banarescu et al., 2013), are an effort to distill the semantic information of text to a graph, and then transform operations mentioned in the text into graph operations involving entities and their relations. Many existing studies have shown the benefits of this transformation across a variety of natural language processing (NLP) tasks, such as paraphrase detection (Issa et al., 2018), machine translation (Song et al., 2019), event extraction (Garg et al., 2015), code generation (Yin and Neubig, 2017), and many others (Dohare and Karnick, 2017; Jangra et al., 2022; Wolfson et al., 2020; Kapanipathi et al., 2021). In Figure 1, we draw an analogy between Arabic numerals, which have greatly improved mathematical representations and made calculations much more convenient, and AMR, which aims to enhance language representation and simplify reasoning over language.

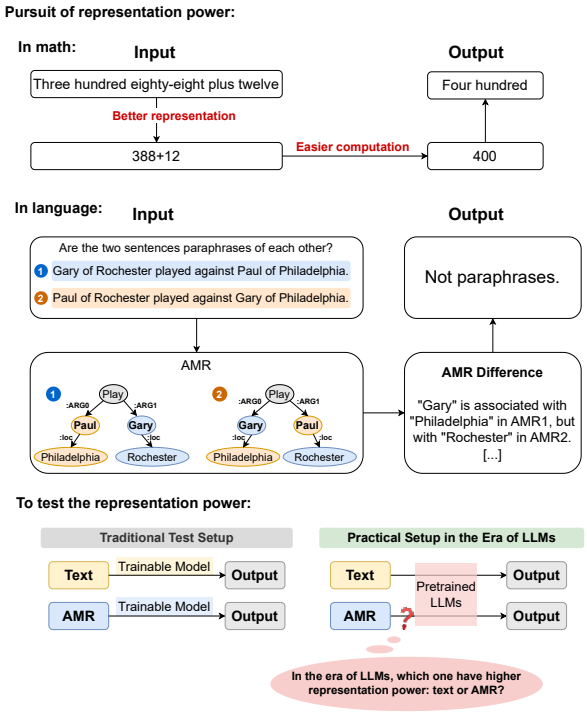

Figure 1: Many fields have long been looking for symbolic languages with better representation power. Analogous to Arabic numbers to math, AMR is usually considered a better representation for text, because for many tasks, the AMR of the sentence makes the computation to the output easier, as shown in the upper half of this figure. Existing work tests the representation power given the assumption that the models are trainable to optimize for the given representation (text or AMR), whereas we look into the practical setup given fixed pre-trained LLMs, as shown in the bottom part of the figure. Our key research question is in the red bubble.

However, although it is a meaningful research direction to explore and evaluate various representations for language (for which we can train new models specific to the proposed representation), there

[1] Our code has been uploaded to the submission system, and will be open-sourced upon acceptance.

might be a different situation when it comes to practical aspects nowadays. Due to economic concerns, there is a growing trend to utilize readily available pre-trained large language models (LLMs) in various application scenarios, without allocating additional resources for training or fine-tuning the models. This shift towards outsourcing model training presents a significant challenge for the semantic representations community, leading to the question:

> *What is the role of semantic representations in the era of LLMs, when no training is involved?*

Motivated by this question, refers to propose the theoretical formulation of representation power, and what it means by an ideal representation for text. We highlight that the key to this paradigm shift is that an ideal representation for language requires optimization of the model with regard to the representation; however, after fixing the LLM, the optimal representation becomes one with regard to this fixed model, which might shift away from the theoretically optimal representation for language. In short, an ideal representation for language does necessarily not mean an ideal representation *with regard to LLMs*.

Paired with the theoretical formulation, we conduct empirical research to understand how good AMR is as a representation in the era of LLMs. Specifically, we answer the following three questions: (1) Does AMR as representation help LLM performance? (2) On what text data does AMR help and what not? (3) Why does it help or not?

We conduct our studies on a diverse set of five NLP tasks, highlighting our findings below. The experiments show that, overall, the contribution of AMR in the era of LLMs is not as high as that in the traditional setup where we can optimize the model for the representation. AMR causes a slight fluctuation of performance by -3 to +1 percentage points. However, we found that AMR is helpful for a large portion of samples, especially those with richer sentence structures such as more adjuncts, and adjective words, as well as a AMR graph with more depth. Finally, we also find that parser-generated AMR leads to a reasonable performance, therefore pushing for more accurate AMR annotations is not the main aspect to improve when using AMR in the era of LLMs. Instead, a potential aspect to improve might be enhancing the chained step-by-step reasoning when LLMs first reason over AMR buy a good understanding of what all its symbols mean, and then need to consistently translate the reasoning on AMR back to the task.

In summary, the contributions of our work are as follows:

1. We are the first to investigate the role of semantic representations in the era of LLMs, especially looking at the practical situation where no training is involved;
2. We propose both a thinking framework to formulate the representation power for language, and comprehensive experimental studies investigating the *whether*, *when*, and *why* behind the role of AMR to LLMs;
3. We reveal a set of findings meaningful for the NLP community to reflect on the contribution of traditional linguistic structures such as semantic representations in the current wave of LLMs, and point out potential areas to improve the performance.

2 A Thinking Framework Formalizing the *Representation Power*

In this section, we propose a unified thinking framework to formulate representation power both in the pre-LLM era, where we do not outsource the training of the models, and the LLM era, where a lot of practical settings are to optimize the representation with regard to a given fixed LLMs.

2.1 Notations

Suppose we have a dataset $D := \{(x_i, y_i)\}_{i=1}^{N}$ consisting of N pairs of input x_i and corresponding output y_i. Given the task to learn the $x \mapsto y$ mapping, we can consider it as a two-stage modeling process: the first step is to convert the raw input x into a good representation r by the representation model $M : x \mapsto r$, and the second step is to do the computation that takes the representation r and predicts the output y by a computation model $C : r \mapsto y$. In this way, we decompose the resulting overall $x \mapsto y$ modeling process into

$$p(y|x) = p(r|x)p(y|r) \ . \quad (1)$$

2.2 Finding the Ideal Representations for Text

Let us take the math example in Figure 1 as a source of inspiration. The first representation r_1 is the English expression such as “Three hundred

eighty-eight plus twelve," and the second representation r_2 is the same calculation in Arabic numbers, "388+12." A characteristic for the representation r_2 to be better than r_1 is that the computation for $C_2 : r_2 \mapsto y$ is much "simpler" than the other function $C_1 : r_1 \mapsto y$. Here, this simplicity notion is over a function, for which the formal measurement is called *Kolmogorov complexity*, or *algorithmic entropy* (Solomonoff, 1964; Kolmogorov, 1965), which is a theoretical construct of the complexity of an algorithm in bits. Intuitively, we can imagine that if we write a computer algorithm to take "Three hundred eighty-eight plus twelve" as input and "Four hundred" as output, this algorithm should be much longer than the other one taking "388+12" as input and "400" as output, since the former would involve more complicated string manipulation to achieve the same effect.

We also use this notion of Kolmogorov complexity to quantify the power of representations for language. Suppose the computation model C can be any from a hypothesis space $\mathcal{C}$, the ideal representation r^* should satisfy

$$r^* = \operatorname*{argmin}_r \operatorname*{argmin}_{C \in \mathcal{C}} K(C : r \mapsto y) \,, \quad (2)$$

given the optimal computation model C with the minimal Kolmogorov complexity K.

If a function has low Kolmogorov complexity, it usually results in several good properties, such as that learning C requires fewer data samples, has smaller empirical risks, and results in more robustness.

This explains the common framework to show AMR as a better representation than the raw text sequence by demonstrating its better performance (Turian et al., 2010), fewer data (Liu et al., 2021), or better robustness and domain transferability (Li et al., 2016). A property of these studies is that they train models customized explicitly for the AMR representation, namely running the optimization of C over the hypothesis space $\mathcal{C}$.

2.3 Representation Power in the Era of LLMs

As mentioned previously, in the era of LLMs, we are moving towards the paradigm where the model is usually fixed, and thus the optimization of the representation is

$$r^*_{\text{LLM}} = \operatorname*{argmin}_r K(C_{\text{LLM}} : r \mapsto y) \,, \quad (3)$$

where we cannot have double optimization of the best model C over the hypothesis space $\mathcal{C}$ for each r, but rather take the existent C_{LLM}, and find the best representation for it. Note that this framework can also be used to to explain the success of chain-of-thought (CoT) prompting in terms of how the new representation generated by CoT unleashes the power of LLMs better.

Comparing Eq. (2) and Eq. (3), we can see that the ideal best representation r^* does not necessarily equal the representation r^*_{LLM} that works well with LLMs, so there remain needs for experiments to fill in this gap of knowledge.

2.4 Additional Practical Concerns

From Eq. (1), we can also see that, when decomposing to this two-stage process, it is likely to accumulate cascading error from the representation step $p(r|x)$, where we might not have an oracle text-to-AMR model, to the computation step $p(y|r)$, which needs to be base on that representation. We make special efforts to address this concern by additional experiments later in section Section 6.1.

3 Designing the AMRCoT Experiments

3.1 Dataset Setup

We include a diverse set of tasks and datasets, spanning across paraphrase detection (Zhang et al., 2019), machine translation (Bojar et al., 2016), logical fallacy detection (Jin et al., 2022), event extraction (Garg et al., 2015), and text-to-SQL generation (Yu et al., 2018). We describe in Table 1 an overview of the tasks and datasets. For each dataset, the size that is reasonable for querying LLMs is several thousand, which is within a reasonable computational budget and can be relatively large enough to make the results representative. To compile the test data for our study, we first take the entire test set of the original data, and if the size is not large enough, we also add the development and training set. The resulting statistics of the data is in Table 1. See the evaluation metrics and setup for each dataset in Appendix A.2.

3.2 AMRCoT Prompt Design

As our study is interested in the representation power in the era of LLMs, we explore how well AMR as the representation works with pre-trained LLMs. We draw inspirations from the CoT prompt design, which enable models to answer an origi-

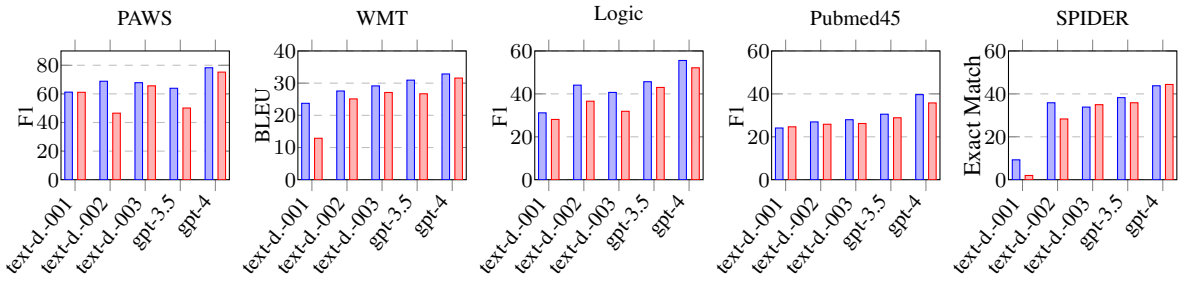

Figure 2: Performance of BASE (in purple) and AMRCOT (in red) on five datasets across the five model versions, from text-davinci-001, -002, -003, to GPT-3.5 and GPT-4.

Dataset	Task	Test Size
PAWS	Paraphrase Detection	8,000
WMT	Translation	5,999
Logic	Logical Fallacy Detection	2,449
Pubmed45	Event Extraction	5,000
SPIDER	Text2SQL Code Generation	8,034

Table 1: Statistic of each dataset, their corresponding task, and the test data size for our study later.

nally difficult question with the help of assistive representations that makes the task easier.

In this work, we propose AMRCOT, which supplements the textual input with its AMR, and conditions the answer generation based on the semantic representation which should make the task easier.

BASE	Please classify the following text into one of the logical fallacies: Text: `{sentence1}` Which is the fallacy type present in the text?
AMRCOT	You are given a text and its AMR. Text: `{sentence1}` **AMR: `{amr1}`** *Based on the text and its AMR*, please classify it into one of the logical fallacies. Which is the fallacy type present in the text?

Table 2: Example BASE and AMRCOT prompt (on the logical fallacy dataset).

In our experiments, we contrast AMRCOT with the baseline of directly querying the LLMs, denoted as BASE. We show an example prompt pair in Table 2, and list all the prompts for all datasets in Appendix A.1.

3.3 Language Models

Since our experiments require models that can reasonably understand the symbols in AMRs as well as reason over them, we find that only the instruction-tuned GPT models, from text-davinci-001 to GPT-4 are capable of doing it, but not the open-sourced models such as LLaMa and Alpaca, at the time we conducted our research. For reproducibility, we set the text generation temperature to 0 for all the models, and we use the model checkpoints on June 13, 2023 for GPT-3.5 and GPT-4, namely `gpt-3.5-turbo-0613` and `gpt-4-0613`.

3.4 Research Question Overview

With all the experimental setup, we explore the following three questions to comprehensively address the role of semantic representations in the era of LLMs: First, can AMR help LLMs when performing different tasks? Second, when does it help/harm and when not? Third, why is it so? We explore each question with three substudies, which we will introduce in the following sections.

4 Q1: Does AMR Help LLMs?

First, we are interested in the overall representation power of semantic representations like AMR to LLMs. Specifically, we answer the following subquestions: what is the overall effect of AMR as a representation on LLMs' performance? (Section 4.1), does the effect vary case by case? (Section 4.2), and how does the effect change on different LLMs with different levels of capabilities? (Section 4.3)

4.1 Overall Effect of AMR

We first look into the overall effect of AMR as a representation to assist LLMs. Table 3 compares the performance of the BASE method of directly querying LLMs and the AMRCOT method, we can see that AMR does not have an overall positive impact on the performance of LLMs. In most of the cases, it makes the performance fluctuate a bit, between a slight increase (e.g., +0.61 in the case of Text-to-SQL code generation) and slight drop (e.g., -1 to -3 in most other tasks).

Dataset	Task	BASE	ΔAMRCOT
PAWS	Paraphrase Detection	78.25	-3.04
WMT	Translation	32.85	-1.27
Logic	Fallacy Detection	55.56	-3.4
Pubmed45	Event Extraction	39.65	-3.87
SPIDER	Text2SQL	43.78	0.61

Table 3: Performance of BASE and AMRCOT on all the five tasks using GPT-4.

4.2 Helpfulness of AMR in Some Cases

There could be several possibilities behind the fluctuating performance: (1) AMR does not add much beyond the plain text input for LLMs, or (2) AMR affects a large number of samples, just that the positive effect and negative effect cancels out each other, so we do not see a big change on the overall statistics.

To explore which of the two actually happens behind the scene, we calculate the "flip rate" before and after using the AMRCOT. In Table 4, we count a sample as helped by AMR if its prediction improves (i.e., changing from incorrect to correct for classification tasks, or having a score increase for text generation tasks), and hurt if its prediction changed vice versa; the rest of the samples are considered unchanged.

Dataset	% Helped	% Hurt	% Unchanged
PAWS	16.48	20.16	63.36
WMT	44.75	47.28	7.97
Logic	22.33	26	51.67
Pubmed45	4.84	11.66	83.5
SPIDER	4.94	4.33	90.72

Table 4: Percentage of samples that are helped (% Helped), hurt (% Hurt), or unchanged (% Unchanged) when we change from BASE to AMRCOT using GPT-4.

As we can see in Table 4, AMR causes a high flip rate on the correctness of the answers. The percentage of answers improved by AMRCOT occupies 44.75% in WMT data, 22.33% in Logic, and 16.48% in PAWS. These are impressive statistics to see, and the remaining question is what features on these text samples causes the performance improvement or drop by AMR, which we will explore in the following Sections 5 and 6.

4.3 AMR's Effect on Models with Different Capabilities

We also add an additional experiment to extend the observation in this section to more models. We show in Figure 2 the performance of BASE and AMRCOT across five LLMs with different levels of capability: from text-davinci-001, -002, -003, to GPT-3.5 and GPT-4.

We can see that, overall, this fluctuation phenomenon exists across all models. Note that in some cases, the less capable models experience a higher drop by AMR, which might be due to its limited ability to comprehend AMR and do reasoning over these special symbols. This is consistent with our other observation that none of the non-instruction-tuned earlier GPT models, and the less capable models such as LLaMa and Alpaca, can comprehend AMR representations well.

5 Q2: When Does AMR Help/Hurt?

The previous section shows that AMR is helpful or harmful for different samples. Now we continue to decode the condition when the help or harm happens, and attribute it to features of the input text. In this section, we first illustrate a case study in Section 5.1, where the lack of the ability of AMR to capture the semantic equivalence of multi-word expressions (MWEs) hinders paraphrase detection. Then, beyond proposing each hypothesis and verifying it for each single phenomenon on each task, we suggest two systematic interpretability studies: for the first method, we treat linguistic features as our hypotheses, and extract features with high correlation with AMR helpfulness in Section 5.2; and for the second method, we directly train classifiers to learn AMR helpfulness as a task, and decode what text input leads to it in Section 5.3.

5.1 Case Study: AMR's Shortcoming on MWEs

AMR has its unique advantages and limitations, from which we can interpret what cases it can help, and what cases not. One such limitation of AMR is its lack of ability to capture MWEs such as slangs, which makes it overlook certain semantic equivalence for paraphrase detection. As in the example below, if an input sentence of paraphrase detection has an MWE, such as "Her *swan song* disappointed her fans.", where the proper of paraphrase for the MWE *swan song* is not "bird song," but "final performance." However, when we show the AMRs for all the three sentences below, we can see that the wrong pair ("swan song" and "bird song") looks more like a paraphrase relation, compared with the correct pair ("swan song" and "final performance").

Given this intuition, we quantitatively verify this hypothesis by running AMRCOT on a self-composed dataset of paraphrase detection involv-

Original Sentence with MWE
*Her **swan song** disappointed her fans.*
(z0 / disappoint-01
:ARG0 (z1 / **song**
:mod (z2 / **swan**)
:poss (z3 / she))
:ARG1 (z4 / fan
:poss z3))

Paraphrase Candidate 1
(✗ Not a paraphrase.)

*Her **bird song** disappointed her fans.*
(z0 / disappoint-01
:ARG0 (z1 / **song**
:mod (z2 / **bird**)
:poss (z3 / she))
:ARG1 (z4 / fan
:poss z3))

Paraphrase Candidate 2
(✓ A paraphrase.)

*Her **final performance** disappointed her fans.*
(z0 / disappoint-01
:ARG0 (z1 / **perform**-02
:ARG0 (z2 / she)
:mod (z3 / **final**))
:ARG1 (z4 / fan
:poss z2))

ing slangs. Since our experiments need annotations for both slang paraphrase pairs and AMRs, we compose two datasets, GoldSlang-ComposedAMR, and GoldAMR-ComposedSlang. For GoldSlang-ComposedAMR, we use the curated slang paraphrase pairs by Tayyar Madabushi et al. (2021), and generate their AMRs with an off-the-shelf parser (Drozdov et al., 2022). For the other dataset, GoldAMR-ComposedSlang, we use gold AMRs from the LDC AMR 3.0 corpus (Banarescu et al., 2013), and compose slang paraphrases using a combination of human efforts and assistance from GPT-4. The data curation steps and data statistics are in Appendix B.1.

Dataset	BASE	ΔAMRCOT
GoldSlang-ComposedAMR	86.83	-6.63
GoldAMR-ComposedSlang	77.69	-8.78

Table 5: AMRCOT causes a large drop in performance on slang-involved paraphrase detection data.

We show the evaluation results in Table 5, where AMRCOT causes a large drop of performance compared with the BASE method, which are more substantial than the slight fluctuation of -3 to +1 on general NLP datasets previously in Table 3. It is very likely that, due to the shortcoming of AMR on MWEs, AMRCOT leads to distraction in the input that makes the model perform worse.

5.2 Large-Scale Text Feature Analysis

The case study above provides a precise insight into a special case when AMR does not work. There could be many other reasons out there, some supporting cases where AMR helps, and others explaining cases where AMR hurts. To systematize such analyses, we avoid the manual effort of verifying each hypotheses, but seeks to scale up the study in a data-driven away.

To enable the systematic analysis, we formulate the contribution of AMR as the *AMR helpfulness* score, which is the difference of the performance by AMRCOT vs. BASE. As all the performance score are in percentage, the AMR helpfulness score is usually a number between -100% and 100%, where a negative value means that AMR hurts the performance of the sample, and positive value means that AMR improves the performance of that sample.

And for each text sample, we compose a large-scale comprehensive set of linguistics features, including 139 text features on the text representation, and 4 AMR-related features. For the over 100 features text, we obtain 55 features using the Text Characterization Toolkit (TCT) (Simig et al., 2022), which is specifically designed to facilitate the analysis of text dataset properties, 17 different part-of-speech (POS) tags, 44 dependency tags, and 61 other handcrafted features, which characterizes the semantic and syntactic complexity of the input text, such as the number of arguments vs. adjuncts (Haspelmath, 2014).

Feature Name	Correlation Coefficient
# Named Entities	0.0552
# Adjuncts	0.0387
Adj POS Tag Frequency	0.0382
Max Word Complexity	0.0291
AMR Graph Depth	0.0286

Table 6: The Pearson correlation between each linguistic feature and AMR helpfulness. The top five correlated features are the number of named entities (# Named Entities), the number of adjuncts (# Adjuncts), the frequency of adjectives among all the words (*Adj* POS Tag Frequency), maximum word complexity level by the age of acquisition (Kuperman et al., 2012), and the maximum length of a direct path in the AMR graph (AMR Graph Depth).

We analyze the Pearson correlation between each linguistic feature and the AMR helpfulness score in Table 6. This per-feature analysis is a more fair reflection of the contribution of each feature, regardless of the potential correlation among multiple features. As we can see, AMR is helpful for samples with features such as more named entities, adjuncts, adjective words, difficult words, and the AMR graph with more depth.

5.3 AMR Helpfulness Prediction as a Learning Task

Apart from the correlation analysis of the large set of linguistic features, we also explore a learning approach to treat AMR helpfulness as a prediction task. Since this is a challenging learning task, we set it up as a binary classification, where the positive class is the cases where AMR helps, and the negative class is the rest. Merging all the five datasets together, we have a binary classification dataset of 18,365 training samples, 2,962 development samples, and 2,962 test samples, with positive labels composing 13.85% of the dataset.

Model	F1	Acc	P	R
Random Baseline	22.54	48.95	14.81	47.11
Using Linguistic Features				
Random Forest	31.32	78.12	25.00	41.92
XGBoost	28.32	59.84	17.98	66.67
Ensemble	**33.12**	74.23	23.96	53.61
Using the Free-Form Text Input				
BERT	45.65	77.65	37.02	59.53
RoBERTa	**46.02**	78.70	38.32	57.60

Table 7: Classification performance of various models on AMR helpfulness. We report the F1, precision (P), and recall (R) of the positive class, as well as the accuracy (Acc). See the implementation details of the models in Appendix A.3.

As shown in the classification results in Table 7, classifiers based on the linguistic features achieve an F1 score of up to 33.12%. After utilizing the entire free-form text input, the BERT–based deep learning models achieve an improvement by up to 12.9 points of the F1 score, as well as substantial increases in both precision and recall.

To increase more interpretability, we also run the SHAPley (Fryer et al., 2021) interpretability method, and find that words that signal the existence of clauses tend to have a high importance for the classifier, such as what, how, said, and says.

6 Q3: Why Does AMR Help/Hurt?

After inspecting the conditions *when* AMR helps or hurts the LLM performance, now we move to more in-depth causal analysis of *why* AMR demonstrates the current representation power with regard to LLMs. Specifically, we look into the following subquestions: (1) how does parser-generated AMR work compared with gold AMR? (Section 6.1) (2) what is the representation power of AMR versus text when being ablated? (Section 6.2) And (3) how does AMR help in each step of the reasoning process? (Section 6.3)

The findings in this section are specifically helpful for understanding the key aspects to work on in the future to improve the performance of AMR on LLMs. For example, parser-generated AMR also leads to a reasonable performance, therefore pushing for more accurate AMR annotations is not the main aspect to improve when using AMR in the era of LLMs. Instead, a potential aspect to improve might be the enhancing chained step-by-step reasoning when LLMs first reason over AMR buy a good understanding of what all its symbols mean, and then need to consistently translate the reasoning on AMR back to the task.

6.1 Gold vs. Parser-Generated AMR

Our first investigation is whether there exists cascading error before the CoT process (as introduced in Section 2.4), due to the parser-generated AMR by Drozdov et al. (2022) (whose reported performance is 83% on AMR 3.0) being not perfect enough. *What would the performance have been had the AMRs were perfect?* – This is the question we address in this subsection.

To check this question, we take the gold text-to-AMR pairs with official annotation from the AMR 3.0 dataset (Banarescu et al., 2013). The challenge is to find a downstream task using the same text input as this AMR dataset with gold annotation. To this end, we look into the source data where the text of the AMR data originates from, and identified that it has certain overlap with the OntoNotes 5.0 dataset (Pradhan et al., 2011). Hence, we take the intersection of the AMR 3.0 dataset and this OntoNotes 5.0 dataset, using the widely used named entity recognition (NER) task. In this way, we make a unique contribution aligning the two datasets, and obtain a precious dataset, *AMR-NER*, consisting of 131 samples with both gold AMR and gold NER annotations.

Dataset	BASE	AMR	ΔAMRCoT
AMR-NER	60.51	Gold	+0.03
		Parser	+1.91

Table 8: Model performance on the AMR-NER data using the gold AMR (Gold) and parser-generated AMR (Parser). We report the BASE performance, and the change of performance by AMRCoT (ΔAMRCoT) in terms of F1 scores.

Using this AMR-NER dataset, we are able to compare the performance of gold AMR versus parser-generated AMR on the same task, NER, in Table 8. We can see that both AMRs lead to similar results,

where there is a slight improvement on the performance, echoing with the previous text feature analysis in Section 5.2 that samples with more named entities tend to be helped by AMR more. Interestingly, the parser-generated AMR actually leads to more improvement, which might be due to its more generic symbols and thus inducing fewer infrequent special tokens that LLMs might not be familiar with.

6.2 Ablating the AMR/Text Representation

As introduced to previously in Section 2, AMR and text representations are two different surface forms to express the same semantics. Either is a sufficient representation containing the complete information of the semantics of the text for LLMs to rely on to complete the task, just that they have different representation power.

With this spirit, we conduct an ablation study to check the reliance of LLM performance on each form of representation. To avoid the additional effect from the parsing process, we use the AMR-NER dataset with the gold AMR.

We first test three variations: providing both text and AMR to LLMs leads to 60.54%, providing only text leads to 60.51% performance, and providing only AMR gives 25.58%. This shows that, no matter how the representation power is in the ideal conditions (as introduced in Section 2.2), if limiting to the pre-trained LLMs (as inSection 2.3), to be a better representation, although AMR can help in marginal cases.

We also extend this analysis to more granularity in Figure 3, where we ablate each of text and AMR by every 20% tokens, and show the effect on the task performance. Similarly to the previous observation, we find that adding more text largely increase the performance of LLMs, showing the higher importance of text as a representation to LLMs.

6.3 Checking the Step-By-Step Reasoning

We further conduct a close-up look at the step-by-step reasoning process produced by AMRCoT. We randomly selected 50 samples from the PAWS dataset, and manually annotated the correctness of each step in the reasoning process. Basically, we have (1) the provided AMR in the prompt to LLMs, which we obtain using the structure BART model (Drozdov et al., 2022), with a reported performance of 82.6 SMATCH scores on the AMR

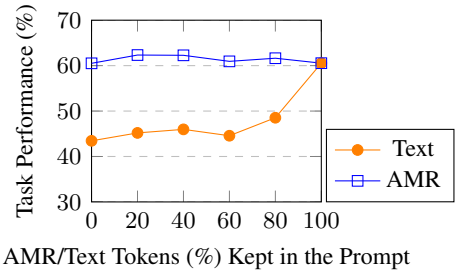

Figure 3: Ablation studies of AMR and text representations in the prompt on the AMR-NER dataset using ChatGPT. Starting from the AMRCoT prompt with the complete text and AMR, we randomly drop out a certain portion of tokens in the text/AMR, and see the effect on the task performance.

3.0 dataset; (2) then for the paraphrase detection task, the LLMs reason about the commonalities and differences between the two AMRs, where we evaluate GPT-4's performance to be 97% F1 scores on listing all the AMR commonalities and differences, with a precision of 95%, and recall of 98%; and (3) finally, the LLM draws conclusion on the paraphrase detection task based on the previous reasoning across the two AMRs, where we evaluate the judgment in this step has a 80% consistency with the reasoning conclusion in step (2).

Overall, one potential reason behind the limitations of AMR for helping the LLM performance could be its several steps in the reasoning are all non-trivially challenging for LLMs, and chaining them together explains why overall AMRCoT achieves a performance of 75.21% on PAWS, which is a slight drop from the BASE performance of 78.25%.

7 Conclusion

In this work, we analyze the role of semantic representations in the era of LLMs. In response to the ongoing paradigm shift in the NLP community, we show that AMR in general is not yet a representation immediately fit for pre-trained LLMs. However, our study show that AMR still help a large portion of text samples, especially those with complicated linguistic structures. We also suggest that a potential direction to enhance AMR's contribution to LLMs is to improve the understanding of LLMs over the schemes and symbols of AMR, and map it to the reasoning of the NLP task. This work presents an effort to bridge the traditionally rich linguistic structures with the strength of LLMs.

Limitations

In this work, we explore one form of linguistic representation of text. And we welcome the methodology in this work to be applied to explore this question on more other linguistic representations.

Ethical Considerations

The datasets used in this paper are existing public datasets on general NLP tasks without any user-sensitive information. We are not aware of specific ethical concerns with the analysis in this study, which is a neutral investigation to understand the role of traditional linguistic structures such as semantic representations in the era of LLMs.

References

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, LAW-ID@ACL 2013, August 8-9, 2013, Sofia, Bulgaria*, pages 178–186. The Association for Computer Linguistics. 1, 6, 7, 11

Ond rej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany. Association for Computational Linguistics. 3

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM. 11

Shibhansh Dohare and Harish Karnick. 2017. Text summarization using abstract meaning representation. 1

Andrew Drozdov, Jiawei Zhou, Radu Florian, Andrew McCallum, Tahira Naseem, Yoon Kim, and Ramón Astudillo. 2022. Inducing and using alignments for transition-based AMR parsing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1086–1098, Seattle, United States. Association for Computational Linguistics. 6, 7, 8, 11

Daniel Vidali Fryer, Inga Strümke, and Hien D. Nguyen. 2021. Shapley values for feature selection: The good, the bad, and the axioms. *CoRR*, abs/2102.10936. 7

Sahil Garg, A. G. Galstyan, Ulf Hermjakob, and Daniel Marcu. 2015. Extracting biomolecular interactions using semantic parsing of biomedical text. *ArXiv*, abs/1512.01587. 1, 3

Martin Haspelmath. 2014. Arguments and adjuncts as language-particular syntactic categories and as comparative concepts. *Linguistic Discovery*, 12:3–11. 6

Fuad Issa, Marco Damonte, Shay B. Cohen, Xiaohui Yan, and Yi Chang. 2018. Abstract meaning representation for paraphrase detection. In *North American Chapter of the Association for Computational Linguistics*. 1

Anubhav Jangra, Preksha Nema, and Aravindan Raghuveer. 2022. T-STAR: Truthful style transfer using AMR graph as intermediate representation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8805–8825, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 1

Zhijing Jin, Abhinav Lalwani, Tejas Vaidhya, Xiaoyu Shen, Yiwen Ding, Zhiheng Lyu, Mrinmaya Sachan, Rada Mihalcea, and Bernhard Schölkopf. 2022. Logical fallacy detection. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7180Ãƒ Â¢Ã¢â€šÂ¬Ã¢â‚¬Å“–7198, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 3

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramón Fernandez Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, Dinesh Garg, Alfio Gliozzo, Sairam Gurajada, Hima Karanam, Naweed Khan, Dinesh Khandelwal, Young-Suk Lee, Yunyao Li, Francois Luus, Ndivhuwo Makondo, Nandana Mihindukulasooriya, Tahira Naseem, Sumit Neelam, Lucian Popa, Revanth Gangi Reddy, Ryan Riegel, Gaetano Rossiello, Udit Sharma, G P Shrivatsa Bhargav, and Mo Yu. 2021. Leveraging Abstract Meaning Representation for knowledge base question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3884–3894, Online. Association for Computational Linguistics. 1

Andrei N Kolmogorov. 1965. Three approaches to the quantitative definition of information. *Problems of information transmission*, 1(1):1–7. 3

Victor Kuperman, Hans Stadthagen-González, and Marc Brysbaert. 2012. Age-of-acquisition ratings for 30,000 english words. *Behavior Research Methods*, 44:978–990. 6

Yitong Li, Trevor Cohn, and Timothy Baldwin. 2016. Learning robust representations of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1979–1985, Austin, Texas. Association for Computational Linguistics. 3

Zhiyuan Liu, Yankai Lin, and Maosong Sun. 2021. Representation learning for natural language processing. *CoRR*, abs/2102.03732. 3

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics. 11

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 shared task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–27, Portland, Oregon, USA. Association for Computational Linguistics. 7

Daniel Simig, Tianlu Wang, Verna Dankers, Peter Henderson, Khuyagbaatar Batsuren, Dieuwke Hupkes, and Mona Diab. 2022. Text characterization toolkit (TCT). In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 72–87, Taipei, Taiwan. Association for Computational Linguistics. 6

Ray J Solomonoff. 1964. A formal theory of inductive inference. part ii. *Information and control*, 7(2):224–254. 3

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using AMR. *Transactions of the Association for Computational Linguistics*, 7:19–31. 1

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics. 6, 11

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden. Association for Computational Linguistics. 3

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *CoRR*, abs/2001.11770. 1

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Vancouver, Canada. Association for Computational Linguistics. 1

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics. 3

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. 3, 12

A Implementation Details

A.1 Prompts

We list the prompts for BASE and AMRCOT of all datasets in Tables 9 and 10, as well as the system prompts in Table 11.

A.2 Evaluation Metrics

For evaluation, we report the performance of PAWS, Logic, and Pubmed45 by F1 scores, the performance of machine translation on the WMT dataset by BLEU scores (Papineni et al., 2002), and the performance of text-to-SQL generation using the official evaluation setup at https://github.com/taoyds/test-suite-sql-eval. To evaluate the generation quality of parser-produced AMRs, we report the SMATCH scores using the evaluation codes at https://github.com/snowblink14/smatch.

A.3 Implementation Details

As for the experimental details, for the BERT and RoBERTa models, we use the weighted cross entropy loss, with a batch size of 16, learning rate of 1e-5, and dropout of 0.1, and train for five epochs until convergence. For the XGBoost classifier (Chen and Guestrin, 2016), we use the default hyperparameters, and set the random seed to 0, and the class weight proportional to the class ratio, namely setting the positive weight to be the inverse of the number of samples in the positive class divided by that of the negative class.

B Data Collection

B.1 Composing the Slang-Involved Paraphrase Detection Dataset

Since our experiments need annotations for both slang paraphrase pairs and AMRs, we compose two datasets, GoldSlang-ComposedAMR, and GoldAMR-ComposedSlang. For GoldSlang-ComposedAMR, we use the curated slang paraphrase pairs by Tayyar Madabushi et al. (2021), and generate their AMRs with an off-the-shelf parser (Drozdov et al., 2022). For the other dataset, GoldAMR-ComposedSlang, we use gold AMRs from the LDC AMR 3.0 corpus (Banarescu et al., 2013), and compose slang paraphrases using a combination of human efforts and assistance from GPT-4.

Composing the GoldSlang-ComposedAMR Dataset We adapt a subset of the ASILM (Tayyar Madabushi et al., 2021), an idiomatic MWE dataset, into a paraphrase detection task. Each sentence in the subset containing idiomatic expressions is paired with a paraphrase (where the idiom is replaced with its literal semantic equivalent) and a non-paraphrase (where the idiom is replaced with a phrase of similar superficial meaning but differing semantic meaning). This results in a balanced paraphrase detection dataset with respect to ground truth labels.

Composing the GoldAMR-ComposedSlang Dataset A possible error in AMRCOT lies in the imperfection of parser-generated AMRs. To disentangle the harm caused by (1) incorrect AMRs produced by the parsers and (2) poor representation of slang expressions by AMRs, we handcrafted the GoldAMR-Slang-Para dataset. We first extract a subset from LDC-AMR3.0 (Banarescu et al., 2013) that involve slang expressions. Then, for each sentence, we replace the slang expression with an alternative expression of the same meaning, and a semantically different expression which seems literally similar, thus creating a paraphrase and non-paraphrase sentence, respectively. The corresponding AMRs can be derived from the original LDC-AMR3.0 AMRs with minimal modifications.

Specifically, we operationalize the process as follows. We first use gpt-3.5-turbo-0613 identify 500 samples of slang usage from LDC-AMR3.0 with the following prompt:

Please evaluate the following sentence for the presence of slang expressions. A slang expression is a phrase or expression that is in the online slang dictionaries and has a meaning that is very different from its literal form. For instance, 'raining cats and dogs' is slang, while 'middle school' is not. Although 'middle school' is a compound phrase, it does not carry a meaning beyond its literal interpretation. Here is the sentence for your analysis: premise. Please format your response as follows: 'Yes or No, slangs.'
If there's no slang used, just answer 'No'. If there are multiple slang expressions, please separate them with a semicolon (';'). Remember, the idioms we are interested in are those that, when taken literally, would have a completely different semantic meaning.

Then we mannually check whether the extracted expressions are slang and are appropriate. Consistent with the spirit of (Zhang et al., 2019), we use the following prompt to query gpt-3.5-turbo-0613 to generate one paraphrase and one non paraphrase of each sentence.

Rewrite the following sentence in two ways Sentence: sentence 1. Replacing "slang" with its intended meaning. 2. Replacing "slang" with its literal meaning, such that the sentence loses its original meaning. Do not change anything else

Lastly, for each pair of (original_sentence, (non)paraphrase_sentence), we give (original_sentence, original_amr, (non)paraphrase_sentence) to gpt-3.5-turbo-0613, and ask it to generate (non)paraphrase_amr by minimally modifying the original_amr. The prompt is as follows:

"The AMR of the sentence 'og_sentence' is og_amr
What is the AMR of the sentence 'paraphrase'? Modified the given AMR to fit the sentence 'hypothesis' and words not present in the sentence 'hypothesis' should not appear in your AMR.
Start you response with '('."

Paraphrase Detection (PAWS)	
BASE	Paraphrase Detection: Determine if the following two sentences are exact paraphrases (rewritten versions with the same meaning) of each other. Sentence 1: `{sentence1}` Sentence 2: `{sentence2}` Answer [Yes/No] and then provide a brief explanation of why you think the sentences are paraphrases or not. Paraphrase:

AMRCOT	Paraphrase Detection: You are given two sentences and the abstract meaning representation (AMR) of each.
	Sentence 1: `{sentence1}`
	AMR 1:
	`{amr1}`
	Sentence 2: `{sentence2}`
	AMR 2:
	`{amr2}`
	Explain what are the commonalities and differences between the two AMRs. Then determine if the two sentences are exact paraphrases (rewritten versions with the same meaning) of each other and provide a brief explanation of why you think the sentences are paraphrases or not. Use the following format: Answer: [Yes/No]

logic	
BASE	Please classify the following text into one of the logical fallacies: Text: `{sentence1}` Which is the fallacy type present in the text?
AMRCOT	You are given a text and its AMR. Text: `{sentence1}` AMR: `{amr1}` Based on the text and its AMR please classify it into one of the logical fallacies. Which is the fallacy type present in the text?
newstest	
BASE	Please translate the following text from English to German. Text: `{sentence1}` Translation:
AMRCOT	You are given a text and its abstract meaning representation (AMR). Text: `{sentence1}` AMR: `{amr1}` Please translate the text from English to German. You can refer to the provided AMR if it helps you in creating the translation. Translation:
pubmed	
BASE	This question aims to assess your proficiency in validating relationships between different entities in biomedical text. You will be presented with a sentence from an article and asked to determine whether the interaction between the entities mentioned in the sentence is valid or not. You should respond with a single digit, either "0" if the interaction is invalid, "1" if it is valid, or "2" if swapping the positions of any two entities would make the interaction valid. Please note that you are required to provide only one of these three responses. Text: `{sentence1}` Interaction: `{interaction}`
AMRCOT	This question aims to assess your proficiency in validating relationships between different entities in biomedical text. You will be presented with a sentence from an article and its abstract meaning representation (AMR) and asked to determine whether the interaction between the entities mentioned in the sentence is valid or not. You should respond with a single digit, either "0" if the interaction is invalid, "1" if it is valid, or "2" if swapping the positions of any two entities would make the interaction valid. Please note that you are required to provide only one of these three responses. Text: `{sentence1}` AMR: `{amr1}` Interaction: `{interaction}`

Table 9: Prompts for BASE and AMRCOT for all datasets.

spider	
BASE	Write an SQL query that retrieves the requested information based on the given natural language question. Remember to use proper SQL syntax and consider any necessary table joins or conditions. Question: `{sentence1}` Query:
AMRCOT	Write an SQL query that retrieves the requested information based on the given natural language question and its abstract meaning representation (AMR). Remember to use proper SQL syntax and consider any necessary table joins or conditions. Question: `{sentence1}` AMR: `{amr1}` Query:
NER	
BASE	The following is a named entity recognition task. Please extract all the named entities of the following types from the given sentence. TYPE="CARDINAL": Numerals that do not fall under another type, e.g., "one", "ten" TYPE="DATE": Absolute or relative dates or periods. E.g., "the summer of 2005", "recent years" TYPE="EVENT": Named hurricanes, battles, wars, sports events, etc. E.g., "Olympiad games" TYPE="FAC": Buildings, airports, highways, bridges, etc. E.g., "Disney", "the North Pole" TYPE="GPE": Countries, cities, states. E.g., "Hong Kong", "Putian" TYPE="LAW": Named documents made into laws. E.g., "Chapter 11 of the federal Bankruptcy Code" TYPE="LOC": Non-GPE locations, mountain ranges, bodies of water. E.g., "Mai Po Marshes", "Asia" TYPE="MONEY": Monetary values, including unit. E.g., "$ 1.3 million", "more than $ 500 million" TYPE="NORP": Nationalities or religious or political groups. E.g., "Chinese", "Buddhism" TYPE="ORDINAL": E.g., "first", "second", etc. TYPE="ORG": Companies, agencies, institutions, etc. E.g., "Eighth Route Army", "the Chinese Communist Party" TYPE="PERCENT": Percentage, including "%". E.g., "25 %" TYPE="PERSON": People, including fictional. E.g., "Zhu De", "Saddam Hussein" TYPE="PRODUCT": Objects, vehicles, foods, etc. (Not services.) E.g., "iPhone", "Coke Cola" TYPE="QUANTITY": Measurements, as of weight or distance. E.g., "23 sq. km" TYPE="TIME": Times smaller than a day. E.g., "homecoming night" Sentence: `{sentence1}` Use json format for the response where each key is an entity type.
AMRCOT	The following is a named entity recognition task. Please extract all the named entities of the following types from the given sentence and its abstract meaning representation (AMR). TYPE="CARDINAL": Numerals that do not fall under another type, e.g., "one", "ten" TYPE="DATE": Absolute or relative dates or periods. E.g., "the summer of 2005", "recent years" TYPE="EVENT": Named hurricanes, battles, wars, sports events, etc. E.g., "Olympiad games" TYPE="FAC": Buildings, airports, highways, bridges, etc. E.g., "Disney", "the North Pole" TYPE="GPE": Countries, cities, states. E.g., "Hong Kong", "Putian" TYPE="LAW": Named documents made into laws. E.g., "Chapter 11 of the federal Bankruptcy Code" TYPE="LOC": Non-GPE locations, mountain ranges, bodies of water. E.g., "Mai Po Marshes", "Asia" TYPE="MONEY": Monetary values, including unit. E.g., "$ 1.3 million", "more than $ 500 million" TYPE="NORP": Nationalities or religious or political groups. E.g., "Chinese", "Buddhism" TYPE="ORDINAL": E.g., "first", "second", etc. TYPE="ORG": Companies, agencies, institutions, etc. E.g., "Eighth Route Army", "the Chinese Communist Party" TYPE="PERCENT": Percentage, including "%". E.g., "25 %" TYPE="PERSON": People, including fictional. E.g., "Zhu De", "Saddam Hussein" TYPE="PRODUCT": Objects, vehicles, foods, etc. (Not services.) E.g., "iPhone", "Coke Cola" TYPE="QUANTITY": Measurements, as of weight or distance. E.g., "23 sq. km" TYPE="TIME": Times smaller than a day. E.g., "homecoming night" Sentence: `{sentence1}` AMR: `{amr1}` Use json format for the response where each key is an entity type.

Table 10: Prompts for BASE and AMRCOT for all datasets.

paws	You are an NLP assistant whose purpose is to perform Paraphrase Identification. The goal of Paraphrase Identification is to determine whether a pair of sentences have the same meaning.
logic	You are an expert in logic whose purpose is to determine the type of logical fallacy present in a text. The categories are: 1) Faulty Generalization 2) False Causality 3) Circular Claim 4) Ad Populum 5) Ad Hominem 6) Deductive Fallacy 7) Appeal to Emotion 8) False Dilemma 9) Equivocation 10) Fallacy of Extension 11) Fallacy of Relevance 12) Fallacy of Credibility 13) Intentional Fallacy.
newstest	You are an NLP assistant expert in machine translation from English to German.
django	You are an NLP assistant expert in translating natural language instructions to python code.
spider	You are a language model designed to generate SQL queries based on natural language questions. Given a question, you need to generate the corresponding SQL query that retrieves the requested information from a database.
pubmed	You are a medical professional expert.
NER	You are an NLP assistant whose purpose is to perform named entity recognition (NER).

Table 11: System prompts for all datasets.