ViAMR: Fine-tuning LLMs for Abstract Meaning Representation in Vietnamese

Dien X. Tran[1,*], Nhon T. Vo[1,*], Kien C. Nguyen[1,*]

[1]Industrial University of Ho Chi Minh City, Vietnam

Corresponding authors.

{22650601.dien, 22658441.nhon}@student.iuh.edu.vn

nguyenchikien@iuh.edu.vn

Abstract

Abstract Meaning Representation (AMR) provides a graph-based view of sentence meaning, typically serialized in PENMAN and evaluated with Smatch. We introduce **ViAMR**, a Vietnamese AMR system that leverages compact open LLMs (Qwen-based decoder-only backbones), fine-tuned through supervised fine-tuning (SFT), and further refines outputs using constrained decoding and lightweight post-processing to produce parsable, well-typed graphs. Our pipeline incorporates structural constraints (balanced parentheses, unique variables, valid role labels), applies rule-based repair to address common Vietnamese-specific errors, and optionally leverages cross-lingual supervision with bilingual inputs and silver data. On the VLSP 2025 shared-task data covering both literary and non-literary domains, ViAMR achieves competitive Smatch with strong parsability and efficiency, ranking among the Top 4 overall while relying only on a small-scale model. We release code, configuration, and evaluation scripts to support reproducibility and future research on Vietnamese AMR.

1 Introduction

Semantic parsing maps natural-language sentences to formal meaning representations that can support reasoning and downstream NLP tasks. Among available formalisms, Abstract Meaning Representation (AMR) (Banarescu et al., 2013) encodes sentence meaning as a directed, rooted graph of concepts and relations, typically serialized in PENMAN notation (Goodman, 2020a). System performance is commonly assessed with Smatch (Cai and Knight, 2013a), which measures graph overlap via precision, recall, and F1.

Research on AMR parsing has advanced rapidly, moving from early discriminative and transition-based models (Flanigan et al., 2014a; Wang et al., 2015a) to neural sequence-to-sequence approaches (Konstas et al., 2017; Zhang et al., 2019), graph-prediction models with latent alignments (Lyu and Titov, 2018), and large pre-trained backbones such as SPRING (Bevilacqua et al., 2021b). More recently, structural guidance and reinforcement-style objectives have been introduced to improve stability and parsability (Yu and Gildea, 2022a).

However, Vietnamese AMR remains under-explored. Gold resources are scarce, spanning multiple domains (literary text, news, reviews), and the language poses unique challenges due to diacritic-rich orthography, productive compounding, and code-switching. These properties often lead to ill-formed graphs (unbalanced parentheses, duplicate variables, or invalid role assignments) and complicate domain generalization.

In this paper, we introduce **ViAMR**, a system for Vietnamese AMR that fine-tunes compact open LLMs using supervised fine-tuning (SFT) and enforces well-formedness through constrained decoding and lightweight post-processing. Our inference pipeline integrates structural constraints (balanced parentheses, unique variables, valid role labels) with rule-based repair for frequent Vietnamese-specific errors. On the VLSP 2025 shared-task data, which covers both literary and non-literary domains, ViAMR achieves competitive Smatch scores and strong parsability, ranking among the Top 4 overall while using only a small-scale model. We also release code and evaluation tools to support reproducibility and further work on Vietnamese AMR.

2 Related Work

In this section, we review prior research in three parts: (i) traditional AMR parsing methods, (ii) modern Transformer/LLM-based approaches, and

(iii) Vietnamese-specific challenges together with our remedies.

2.1 Traditional AMR parsing

The first AMR parsers explicitly assembled graphs through constrained search, enforcing structural validity at every step. JAMR introduced a discriminative, two-stage pipeline (concept identification then relation attachment) and set the first widely adopted baseline (Flanigan et al., 2014b). Transition-based algorithms subsequently mapped dependency trees to AMR via action sequences, improving F-measure while preserving structural control (Wang et al., 2015b). These lines clarified the roles of aligners, graph assembly, and constrained decoding in producing well-formed graphs.

2.2 Modern Transformer and LLM-based approaches

Pretrained Transformers made linearized text↔AMR transduction the default. SPRING attained strong accuracy with symmetric parsing–generation on BART backbones (Bevilacqua et al., 2021a), and AMRBART added graph self-supervised pretraining for stronger structure awareness (Bai et al., 2022). Decoder-side structural signals such as ancestor information further stabilize decoding (Yu and Gildea, 2022b). In parallel, large decoder-only LLMs can produce plausible PENMAN from prompts but lag specialist parsers on semantic correctness and parsability; recent studies report near-zero fully correct parses and substantially lower Smatch even with few-shot/CoT prompting (Ettinger et al., 2023; Li and Fowlie, 2025). These findings motivate task-specific fine-tuning plus lightweight constraints for reliable AMR.

2.3 Cross-lingual AMR and Vietnamese-specific challenges

Because gold AMR resources are concentrated in English, cross-lingual transfer is central for low-resource languages. XL-AMR demonstrates transfer via annotation projection/model transfer (Blloshmi et al., 2020); bilingual input objectives sharpen concept prediction (Cai et al., 2021b); and multilingual noisy knowledge distillation learns a single parser for multiple languages using a strong English teacher (Cai et al., 2021a). Vietnamese adds practical hurdles for linearized AMR multi-syllabic words with diacritics, compounding, and segmentation variability. Therefore, we paired SFT with constraint-aware inference and `penman` round-trip normalization, and we reported both Smatch (Cai and Knight, 2013b) and phenomenon-aware validity metrics (e.g., via GRAPES) (Groschwitz et al., 2023). *Leveraging ViAMR to tame Vietnamese-specific issues.* To operationalize the above challenges, we bake the fixes into the data and supervision itself. Concretely, ViAMR enforces (i) stable surface forms for multi-syllabic words and names by preserving diacritics in text but requiring variables to be lowercase ASCII and replacing spaces with underscores in multiword concepts crucial because in Vietnamese whitespace separates syllables rather than words; this reduces token drift and stabilizes alignments in PENMAN (Vu et al., 2018). (ii) Graph hygiene is standardized directly in the gold strings (balanced parentheses, unique variables, canonical role labels), so the model consistently learns well-formed graphs and avoids brittle formatting; at inference we mirror the same conventions and run a PENMAN round-trip to keep predictions parsable and comparable (Goodman, 2020b). (iii) Domain diversity is retained (literary, news, reviews), and optional Vietnamese pre-segmentation/Named Entity Recognition (NER) hooks remain compatible with common toolkits, which lowers variance when the register shifts while staying model-agnostic. Together, these choices let cross-lingual priors transfer, while the ViAMR conventions absorb Vietnamese-specific noise rather than letting it leak into decoding.

3 Dataset

In this study, we utilize the dataset provided by the VLSP Shared Task on Semantic Parsing. The goal of this task is to build a semantic parser for Vietnamese, enabling accurate understanding and formal representation of sentences by analyzing both their syntactic and semantic structures. Such a parser aims to extract the underlying meaning of text and convert it into structured forms such as Abstract Meaning Representation (AMR) or logical expressions. This resource is especially important in the context of Vietnamese, where large-scale annotated datasets and semantic resources remain limited, and it has the potential to enhance downstream NLP tasks such as machine translation, information extraction, and question answer-

```
# ::snt và em đỏ mặt.
(a / and
    :op2(đ / đỏ
        :pivot(e / em)
        :compound(m / mặt)))

#::snt – để quên nỗi xấu hổ của
ta , bợm nhậu cúi đầu thú nhận .
(t1 / thú_nhận
    :purpose(q / quên
        :theme(n / nỗi
            :compound(x /
            ↪  xấu_hổ)
            :source(t / ta))
        :pivot b)
    :agent(b / bợm
        :compound(n1 / nhậu))
    :manner(c1 / cúi
        :theme(đ1 / đầu
            :part-of b)))
```

Figure 1: Examples of Vietnamese sentences and their AMR graphs (PENMAN format).

Split	# Sentences	Avg. tokens	Avg. chars
Train	1,750	11.54	44.17
Public test	150	17.13	68.24
Private test	1,200	13.20	54.17

Table 1: Corpus statistics of the VLSP dataset.

ing.

Training and Test Data. The organizers provide annotation guidelines together with training and test datasets. The sentences are drawn from multiple domains, including *VietTreebank*, the novel *The Little Prince*, news articles, restaurant and hotel reviews, among others. All data is semantically annotated according to the task schema and organized in *PENMAN* format.

Figure 1 shows examples of Vietnamese sentences and their corresponding AMR graphs. Corpus statistics are summarized in Table 1, including the number of sentences, average number of tokens, and average number of characters. Furthermore, Table 2 reports the most frequent semantic roles in the training graphs, such as `:mod`, `:agent`, and `:theme`, which reflect common syntactic–semantic patterns in Vietnamese. These statistics provide an overview of the linguistic and semantic structures captured in the VLSP dataset.

Role	Count	Role	Count
:mod	1,236	:domain	392
:agent	1,142	:op1	381
:theme	776	:op2	362
:pivot	541	:polarity	299
:compound	534	:time	290
:topic	487	:name	288
:classifier	448	:degree	425
:quant	416	:manner	394

Table 2: Most frequent semantic roles in the training graphs.

4 Methodology

Our methodology is organized into three main components: preprocessing, supervised fine-tuning, and an inference pipeline. In the preprocessing stage, PENMAN-formatted AMR graphs are normalized and repaired to ensure syntactic validity. We then fine-tune a compact decoder-only backbone using supervised fine-tuning (SFT) on the Vietnamese AMR dataset to adapt multilingual representations to Vietnamese-specific AMR structures. Finally, we design a constraint-aware inference pipeline that integrates structural rules and lightweight post-processing, ensuring that the generated graphs are well-formed, parsable, and robust for downstream tasks.

4.1 Preprocessing

Before training and inference, we apply several preprocessing steps to normalize PENMAN-formatted AMR graphs and ensure that they are syntactically valid:

- **One-line normalization**: convert multi-line PENMAN graphs into a single-line format by trimming unnecessary characters and reducing redundant whitespaces. This makes the data more consistent and easier to handle during training.
- **Fixing missing brackets**: count the number of opening and closing parentheses, and automatically append missing closing brackets if an imbalance is detected, thus preserving well-formed graph structures.
- **Multiword node handling**: replace whitespaces in multiword nodes with underscores (e.g., *"xấu hổ"* → *"xấu_hổ"*) ensuring a consistent and parsable representation.

These preprocessing steps reduce common syntactic errors in Vietnamese AMR and improve the robustness of both generation and parsing.

4.2 SFT Fine-tuning

We fine-tune a compact decoder-only backbone using supervised fine-tuning (SFT) on the Vietnamese AMR dataset. The backbone is initialized from the Qwen3-1.7B model (Yang et al., 2025), which provides strong multilingual priors while maintaining efficiency. Each input sentence is paired with its gold AMR graph in PENMAN format, and the model parameters are optimized to maximize the likelihood of generating the correct graph sequence.

To stabilize training and prevent overfitting, we employ the AdamW optimizer with a linear decay schedule and warmup. The maximum sequence length is set to 2048 tokens, with gradient accumulation to simulate larger batch sizes under hardware constraints. Training is performed for 20 epochs with distributed optimization (DeepSpeed ZeRO-2), ensuring scalability and efficient memory usage.

Prompted SFT format. We cast text→AMR as instruction-following causal generation. For each example, the model receives a *system prompt* that codifies output format and role constraints (Table 3), followed by the Vietnamese sentence; the supervision target is the gold PENMAN string. We apply label masking so that loss is computed only on AMR tokens (not on the prompt nor the input sentence), encouraging the model to place a well-formed AMR inside the prescribed delimiters while avoiding spurious learning from instruction text. This "answer-only" masking also makes the causal objective align with downstream inference, where only the AMR is decoded.

The full set of hyperparameters and implementation details for the fine-tuning pipeline is summarized in Table 4.

4.3 Inference Pipeline

Generating a well-formed AMR graph from a raw Vietnamese sentence involves several logical steps. Even with fine-tuned models, direct generation often yields minor formatting or structural glitches (e.g., unbalanced parentheses, merged tokens) that break parsing. We therefore adopt a modular pipeline to ensure robustness without sacrificing semantics. The overall flow is shown in Figure 2.

(1) Test Set Input. We take raw Vietnamese sentences as plain text, preserving diacritics and underscores for multiword concepts. No labels are provided at this stage.

(2) AMR Generate. A decoder-only LLM (Qwen3-1.7B) fine-tuned via SFT produces a linearized AMR in PENMAN notation, following the text-to-AMR paradigm (Bevilacqua et al., 2021b). PENMAN is the standard serialization and has robust parsing/formatting APIs for graph I/O (Goodman, 2020a).

(3) Post-processing (string-level repairs). We apply a fast, deterministic cascade of string-level fixes before parsing the PENMAN string into a graph:

- **Ensure space before roles**: enforce exactly one space before every role label (e.g., `:ARG0`, `:op1`) to avoid token merging that breaks downstream parsing.
- **Fix unmatched parentheses**: count opening/-closing parentheses and append the missing closers to re-establish a well-formed, nested structure required by PENMAN.
- **Fix empty roles**: drop or repair dangling role labels (e.g., a stray `:ARG1` without a following node) that have no semantic target.
- **Fix bare concepts**: normalize nodes to the canonical `<var> / <concept>` form; remove orphan slashes or introduce a fresh variable when needed. For multiword concepts, compact internal spaces into underscores to match AMR conventions.
- **Dedup variables**: detect variable-name collisions and rename consistently (`a0`, `a1`, ...) while updating all references (alpha-conversion; meaning-preserving).

These repairs are inexpensive ($O(n)$ over the string), idempotent, and model-agnostic. They follow best practices in AMR pipelines: use lightweight normalization prior to graph I/O, then rely on the `PENMAN` toolkit for robust parsing/round-trip canonicalization and optional AMR normalization passes for fairer evaluation.

Ordering & safety. We apply repairs in the order above; each step either leaves the string unchanged or makes a local, semantics-preserving fix. This mirrors the "light post-processing" practice reported for production AMR pipelines like SPRING (normalization before evaluation) (Bevilacqua et al., 2021b).

```
You are a large language model specialized in Vietnamese semantic parsing.
Your task is to convert a Vietnamese input sentence into a complete AMR
representation.

Rules:
1. The input is a natural Vietnamese sentence.
2. Think through the analysis and plan the AMR construction inside
<think>...</think>.
3. After finishing the reasoning, output only the final AMR inside
<answer>...</answer>.
4. The AMR must follow PENMAN syntax:
   - Use `(var / concept :role (var2 / concept2) ...)`.
   - Variables are lowercase ASCII (no diacritics).
   - Include semantic roles (:agent, :patient, :location, :time, :mod, :domain,
   etc.).
   - Variables must be unique. Name by the first letter of the concept +
   increasing index
     (e.g., "co" -> c, "chi" -> c1, "cho" -> c2).
   - Ensure each variable represents exactly one concept.
5. No extra newlines/indentation or explanations outside <think> and <answer>.
6. If the sentence cannot be fully analyzed, return a minimal AMR that captures
the main idea.

Example:
Input: "cứ mỗi năm hành tinh này lại quay nhanh hơn , thế mà điều lệnh không
thay đổi !"

Output:
<answer>(c / contrast-01 :ARG1 (q / quay :frequency (n / nam) :theme (h /
hanh_tinh :mod (n1 / nay)) :manner (n2 / nhanh :degree (h1 / hon))) :ARG2 (t1 /
thay_doi :theme (d / dieu_lenh) :polarity -))</answer>
```

Table 3: SFT training prompt used in our system.

(4) AMR Cleanse (graph-level normalization). We parse the repaired string with PENMAN, then immediately perform a decode→encode round-trip to canonicalize spacing and variable naming (Goodman, 2020a). We also enable AMR normalization passes to collapse meaning-equivalent variants so that semantically identical graphs are not penalized due to surface differences.

5 Experimental Setup

5.1 Implementation Details

We use a single-stage *supervised fine-tuning* (SFT) pipeline with a compact decoder-only backbone and a lightweight, constraint-aware inference stack. Our backbone is Qwen/Qwen3-1.7B (dense, 1.7B parameters), which provides strong multilingual priors at a small footprint. Table 4 summarizes training and inference settings.

Configuration	Training
Optimizer	AdamW
Learning rate	1e-5
LR schedule	linear decay (warmup 5%)
Max seq. length	2048
Per-device batch	1
Grad. accumulation	1
Weight decay	0.01
Warmup steps	1000
Epochs	20
Distributed	DeepSpeed ZeRO-2

Table 4: Implementation details for the SFT pipeline on Qwen3-1.7B and the constraint-aware inference stack.

5.2 Evaluation Metrics

We evaluate our fine-tuned model with Smatch, the de facto AMR metric that compares two semantic graphs via *triple overlap* under an optimal variable (node) alignment. For each prediction–gold pair, let M be the maximum number of matching triples, T the number of triples in the prediction, and G the number in the gold graph. Precision, recall, and F_1 are:

$$P = \frac{M}{T}, \qquad R = \frac{M}{G}, \qquad F_1 = \frac{2PR}{P+R}. \quad (1)$$

Unless otherwise stated, we use the official implementation with default hill-climbing restarts and

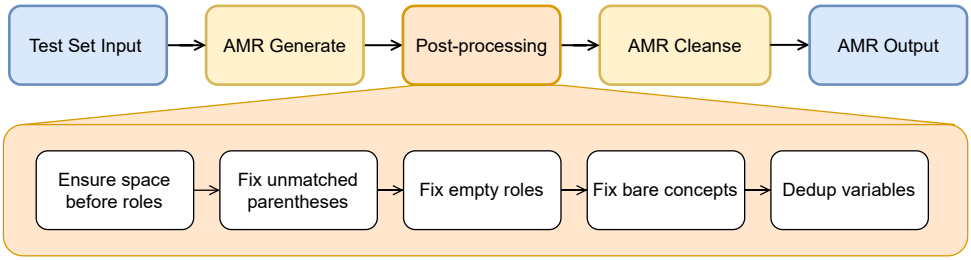

Figure 2: AMR inference pipeline architecture. The highlighted block expands the string-level repair steps we apply before graph parsing.

Field	Value
Split	*private_test* (leaderboard)
Metric	Smatch (macro F_1)
Scoring protocol	One-line PENMAN + round-trip before scoring
System	VIAMR (SFT + constraints)
Inference constraints	Role spacing; underscores; balanced parentheses; alpha-renaming
Result	**F_1 = 0.46**

Table 5: Official leaderboard score and scoring context.

macro-average the sentence-level scores across the corpus.

6 Results and Analysis

6.1 Leaderboard Evaluation and Analysis

On the challenge's hidden *private_test* split, our official submission achieves a macro-averaged Smatch F_1 of **0.46**. We compute Smatch (Cai and Knight, 2013a) on single-line PENMAN after a structure-first normalization that (i) enforces exactly one space before each role label (e.g., `:ARG0`, `:op1`); (ii) repairs unmatched parentheses by inserting the missing closers; (iii) drops or fixes empty roles that lack a target; (iv) normalizes bare concepts to the canonical `<var> / <concept>` form; and (v) alpha-renames variables to deduplicate names while updating all references. We then parse the repaired string into a PENMAN graph and immediately re-serialize it (`decode→encode`) using the `PENMAN` library (Goodman, 2020a) to obtain a canonical surface with consistent spacing/quoting and balanced brackets, yielding parsable, low-variance inputs for Smatch. This "structure-first, then score" practice follows established AMR pipelines such as SPRING (Bevilacqua et al., 2021b). Table 5 summarizes the evaluation context and system setup.

Post-processing (decode→encode). Two-stage pipeline: (1) the string-level repairs (items i–v above), then (2) round-trip canonicalization by `decode` to a PENMAN graph and `encode` back to PENMAN (Goodman, 2020a). This ensures every prediction is parsable and consistently formatted before scoring (Cai and Knight, 2013a).

6.2 Error Analysis

We compare gold–prediction pairs on the public set (aligned by `#::snt`) and group failures along five fine-grained dimensions inspired by the GrAPES evaluation suite for AMR (Groschwitz et al., 2023). The distribution in Figure 3 shows a highly skewed profile: Multiword Concept (Underscore) dominates (83 cases), indicating that multi-syllabic concepts and named entities are frequently rendered without underscores or with altered diacritics, which preserves parsability but harms string-sensitive alignment; Reentrancy (Variable Sharing) is the next major source (16), where repeated mentions are realized as fresh nodes rather than shared variables; Scope (Negation) (9) reflects polarity attached at the wrong predicate level; Attachment (Time/Location) (2) mainly concerns temporal/locative modifiers bound to nominals instead of the event predicate; and Quotation Semantics (3) arises when quoted content is serialized as a literal string under a speech verb instead of a clausal proposition. These are predominantly *semantic* errors rather than surface formatting: our structure-first post-processing with `penman` round-trip eliminates bracket/spacing/collision artifacts before scoring (Goodman, 2020a), but it cannot correct role choice, variable sharing, or scope. This motivates structure-aware supervision and decoding—for example, injecting ancestor/constraint signals at generation time to stabilize role/scopal decisions and recover reentrancies (Yu and Gildea, 2022a). Extended Vietnamese/English examples

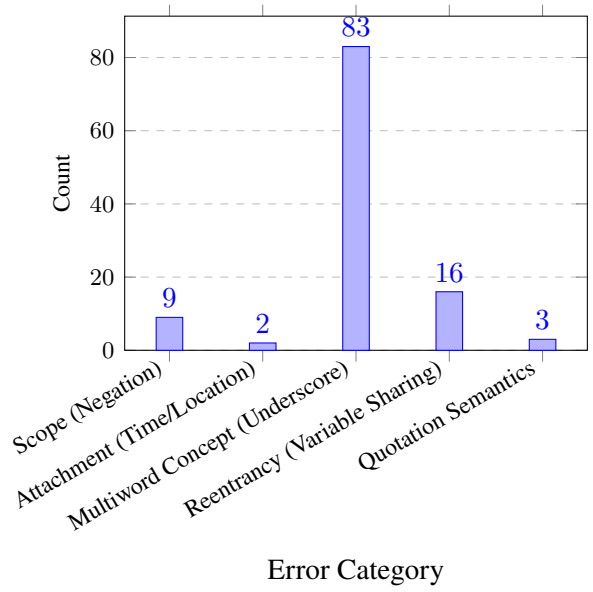

Figure 3: Distribution of error categories (detector-based)

with side-by-side PREDICT vs. GOLD AMR are provided in Appendix A.

7 Discussion

Our results support a simple recipe for Vietnamese AMR: *structure first, semantics next*. A constraint-aware *post-processing* stack (i) enforces a single space before each role label, (ii) underscore multiword concepts, (iii) balance parentheses, (iv) drop or fix empty roles and bare concepts, and (v) alpha-rename variables followed by a PENMAN `decode→encode` round trip consistently produced parsable, low-variance outputs and stabilized Smatch on *private_test*. Within this stable I/O frame, supervised fine-tuning of a compact decoder-only backbone (Qwen3-1.7B) was more reliable than prompt-only decoding: answer-only masking yielded well-formed PENMAN, and greedy/short-beam decoding plus the same post-processing kept inference cheap. Remaining errors are largely semantic (role confusions, long-distance reentrancies/coreference, domain shift). Two promising directions are (i) stronger structural signals during learning (e.g., teacher guidance or constrained decoding with ancestor/role schemas) and (ii) reinforcement-style fine-tuning with AMR-aware rewards that combine Smatch and parsability, building on the stable `decode→encode` interface.

8 Conclusion

We introduced VIAMR, a compact, reproducible Vietnamese AMR pipeline that combines supervised fine-tuning with a constraint-aware inference stack. Simple but disciplined *post-processing* enforcing a single space before each role label, using underscores for multiword concepts, balancing parentheses, dropping or fixing empty roles and bare concepts, and alpha-renaming variables together with a PENMAN `decode→encode` round trip consistently yielded parsable graphs and stable evaluation across domains, delivering competitive Smatch scores with modest computational cost and predictable latency.

In the future, we will introduce a *teacher–student data engine* built on GPT-4o for automatic curation and routing. Specifically, we will use GPT-4o as a *reasoning teacher* to generate an initial `<answer>` AMR for each Vietnamese input. If this `<answer>` matches the gold AMR, we aggregate these examples into the *SFT-positive* set to train instruction-tuning with reasoning. Conversely, if the `<answer>` differs from the gold AMR, we route such examples into the *GRPO-hard* set for reinforcement-style training, optimized with Group Relative Policy Optimization (GRPO). This approach enables us to both expand clean supervision data and focus on hard cases through policy optimization.

References

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for amr parsing and generation. In *Proceedings of ACL 2022 (Long Papers)*, pages 6001–6015, Dublin, Ireland. Association for Computational Linguistics.

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

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021a. One spring to rule them both: Symmetric amr semantic parsing and generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12564–12573.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021b. One spring to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *AAAI*.

Rexhina Blloshmi, Rocco Tripodi, and Roberto Navigli. 2020. Xl-amr: Enabling cross-lingual amr parsing with transfer learning techniques. In *Proceedings of EMNLP 2020*, Online. Association for Computational Linguistics.

Deng Cai, Xin Li, Jackie Chun-Sing Ho, Lidong Bing, and Wai Lam. 2021a. Multilingual amr parsing with noisy knowledge distillation. In *Findings of EMNLP 2021*, pages 2778–2789, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013a. Smatch: an evaluation metric for AMR. In *Proceedings of ACL 2013 (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013b. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of ACL 2013 (Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Yitao Cai, Zhe Lin, and Xiaojun Wan. 2021b. Making better use of bilingual information for cross-lingual amr parsing. In *Findings of ACL-IJCNLP 2021*, pages 1537–1547, Online. Association for Computational Linguistics.

Allyson Ettinger, Jena D. Hwang, Valentina Pyatkin, Chandra Bhagavatula, and Yejin Choi. 2023. "you are an expert linguistic annotator": Limits of llms as analyzers of abstract meaning representation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8250–8263, Singapore. Association for Computational Linguistics.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014a. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of ACL 2014 (Volume 1: Long Papers)*, pages 1426–1436, Baltimore, Maryland. Association for Computational Linguistics.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014b. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of ACL 2014 (Long Papers)*, pages 1426–1436, Baltimore, Maryland. Association for Computational Linguistics.

Michael Wayne Goodman. 2020a. Penman: An AMR graph library for python. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Online. Association for Computational Linguistics.

Michael Wayne Goodman. 2020b. Penman: An open-source library and tool for amr graphs. In *Proceedings of ACL 2020: System Demonstrations*, pages 312–319, Online. Association for Computational Linguistics.

Jonas Groschwitz, Matthias Lindemann, Axel Nyström, Simon Petitjean, Djamé Seddah, Jakub Waszczuk, and Stephan Oepen. 2023. Amr parsing is far from solved: Grapes, the granular amr parsing evaluation suite. In *Proceedings of EMNLP 2023*, pages 10728–10752, Singapore. Association for Computational Linguistics.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of ACL 2017 (Volume 1: Long Papers)*, pages 1173–1183, Vancouver, Canada. Association for Computational Linguistics.

Yanming Li and Meaghan Fowlie. 2025. Gpt makes a poor amr parser. *Journal for Language Technology and Computational Linguistics*, 38(2):43–76.

Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of ACL 2018 (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia. Association for Computational Linguistics.

Thanh Vu, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. 2018. Vncorenlp: A vietnamese natural language processing toolkit. In *Proceedings of NAACL 2018: Demonstrations*, pages 56–60, New Orleans, Louisiana. Association for Computational Linguistics.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015a. A transition-based algorithm for AMR parsing. In *Proceedings of NAACL-HLT 2015*, pages 366–375, Denver, Colorado. Association for Computational Linguistics.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015b. A transition-based algorithm for amr parsing. In *Proceedings of NAACL-HLT 2015*, pages 366–375, Denver, Colorado. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 technical report.

Chen Yu and Daniel Gildea. 2022a. Sequence-to-sequence AMR parsing with ancestor information. In *ACL (Short)*.

Chen Yu and Daniel Gildea. 2022b. Sequence-to-sequence amr parsing with ancestor information. In *Proceedings of ACL 2022 (Short Papers)*, pages 571–577, Dublin, Ireland. Association for Computational Linguistics.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. Amr parsing as sequence-to-graph transduction. In *Proceedings of ACL 2019 (Volume 1: Long Papers)*, pages 80–94, Florence, Italy. Association for Computational Linguistics.

A Extended Error Examples

We provide side-by-side Vietnamese/English sentences with concise AMR fragments from the LLM predictions and the gold references (PENMAN via Goodman (2020a); error dimensions follow Groschwitz et al. (2023)). A consolidated sheet of representative cases is shown in Figure 4;

Category	Sentence	Predict	Gold
Scope (Negation)	VN: “anh ấy không hề phủ nhận rằng mình đã sai, nhưng cũng không chấp nhận xin lỗi công khai vì cho rằng hoàn cảnh lúc đó quá phức tạp để giải thích rõ ràng.” EN: “He does not admit that he was wrong, but he also refuses to apologize publicly, arguing that the circumstances then were too complex to explain clearly.”	(c / contrast-01) :ARG1 (a / admit-01 :ARG0 (h / he) :ARG1 (w / wrong-05 :ARG1 h)) :ARG2 (r / refuse-01 :ARG0 h :ARG1 (a2 / apologize-01 :manner (p / public))) :polarity -	(c / contrast-01 :ARG1 (a / admit-01 :ARG0 (h / he) :ARG1 (w / wrong-05 :ARG1 h) :polarity -) :ARG2 (r / refuse-01 :ARG0 h :ARG1 (a2 / apologize-01 :manner (p / public)) :ARG2 (a3 / argue-01 :ARG0 h :ARG1 (c2 / complex-03 :domain (c3 / circumstance :time (t / then)) :degree (t2 / too)) :ARG2 (e / explain-01 :manner (c4 / clearly))))
Attachment (Time/Location)	VN: “cuộc hội thảo quốc tế về đô thị thông minh diễn ra vào sáng thứ Hai tuần tới tại Đại học Quốc gia Hà Nội, kéo dài đến chiều và kết thúc bằng một phiên thảo luận mở.” EN: “The international workshop on smart cities takes place next Monday morning at Vietnam National University, Hanoi, lasts until the afternoon, and ends with an open discussion session.”	(h / hoi_thao :mod (q / quoc_te) :topic (d / do_thi_thong_minh) :time (s / sang :mod (t / thu_hai :time (t1 / tuan_toi))) :location (v / VNU_Ha_Noi) :duration (c / chieu) :manner (p / phien_thao_luan :mod (m / mo)))	(d / dien_ra-01 :ARG1 (h / hoi_thao :mod (q / quoc_te) :topic (d /do_thi_thong_minh)) :time (s / sang :time (t / thu_hai :time (t1 / tuan_toi))) :location (v / VNU_Ha_Noi) :duration (u / until :op1 (c / chieu)) :concession (k / ket_thuc-01 :ARG1 h :manner (p / phien_thao_luan :mod (m / mo))
Multiword Concept (Underscore)	VN: “cậu bé rất xấu hổ.” EN: “The boy is very embarrassed.”	(e / embarrassed-01) :ARG0 (b / cau_be) :manner (x / xau) :mod (h / ho)	(e / embarrassed-01) :ARG0 (b / cau_be) :manner (x / xau_ho)
Reentrancy (Variable Sharing)	VN: “Lan gặp Hùng và Lan chào Hùng.” EN: “Lan met Hung and Lan greeted Hung.”	(a / and) :op1 (g / gap-01 :ARG0 (l / Lan) :ARG1 (h / Hung)) :op2 (c / chao-01 :ARG0 (l / Hung) :ARG1 (h / Lan)	(a / and) :op1 (g / gap-01 :ARG0 (l / Lan) :ARG1 (h / Hung)) :op2 (c / chao-01 :ARG0 l :ARG1 h)
Quotation Semantics	VN: “ông nói: ‘tôi sẽ trở lại.’” EN: “He said: ‘I will return.’”	(s / say-01) :ARG0 (o / ong) :ARG1 "toi se tro lai"	(s / say-01) :ARG0 (o / ong) :ARG1 (r / return-01 :ARG0 (t / toi))

Figure 4: Representative error cases (VN ↔ EN, PREDICT vs. GOLD). Categories include *Scope (Negation)*, *Attachment (Time/Location)*, *Multiword Concept (Underscore)*, *Reentrancy (Variable Sharing)*, and *Quotation Semantics*. The sheets highlight where PREDICT diverges from GOLD and how these differences impact Smatch.