Sema System for the DMR 2026 Shared Task: Multistage UMR Parsing with Qwen3-4B

Rémi de Vergnette, Maxime Amblard
LORIA, UMR 7503, Université de Lorraine, CNRS, Inria
54000 Nancy, France
remi.de-vergnette@loria.fr, maxime.amblard@univ-lorraine.fr

Abstract

We present the Sema system for the DMR 2026 shared task on parsing from natural language to UMR. Our approach relies on parameter-efficient fine-tuning of Qwen3-4B with a multistage training procedure. We first train on a capped subset of the noisy training data, then continue training on the clean split, and finally fine-tune a dedicated stage for word-to-node alignment prediction. The system generates sentence-level graphs, selected document-level information, and alignments in separate steps, followed by rule-based post-processing to satisfy the official evaluation format. Results show that the approach is viable across several languages and exhibits promising transfer to Italian despite the absence of Italian data for fine-tuning , while very low-resource languages remain challenging.

Introduction

Parsing to AMR (Banarescu et al., 2013) has been studied extensively. In contrast, parsing from natural language to UMR (Bonn et al., 2024) remains much less explored. UMR extends AMR with the goal of providing a more expressive representation of meaning and capturing phenomena that are not well handled in English-centered sentence-level graph formalisms. Compared with AMR, UMR differs in three main ways: (i) it annotates phenomena that are largely ignored in AMR, (ii) it explicitly includes node-to-word alignment, and (iii) it introduces an additional document-level layer to capture inter-sentential phenomena.

The DMR 2026 shared task (Štěpánek et al., 2026) is the first shared task dedicated to this problem. In this paper, we present the system we developed for this shared task. Our system is based on parameter-efficient fine-tuning (PEFT) of a large language model (LLM) in several stages. More precisely, it combines four main design choices: staged fine-tuning on noisy and clean data, a dedicated alignment generation stage, a custom linearization for alignments, and rule-based post-processing to satisfy the official evaluation format. Contrary to previous work on UMR parsing (Chun and Xue, 2024; Markle et al., 2026), we do not explicitly rely on AMR as an intermediate step by either training on AMR data or using an AMR parser. Instead, we directly train on UMR data.

The remainder of the paper is organized as follows. Section 1 describes the data. Section 2 presents the system architecture and training procedure. Section 3 describes generation. Section 4 details the post-processing pipeline. Section 5 gives official results and some elements of analysis.

	clean	dirty
English	180	29872
Chinese	557	1435
Navajo	5	337
Arapaho	53	292
Czech	103	159906
Latin	0	1049

Table 1: Number of training samples per split and language.

1. Data

The DMR 2026 shared task provides a training set composed of two splits: a *dirty* split and a *clean* split. The number of samples of each split, for each language, is given in Table 1. The training data is highly unbalanced, especially in the dirty split, where Czech represents more than 80% of the samples. Annotation quality also varies across the data. In particular, the dirty split contains automatically generated annotations that may be incomplete, especially for the document-level layer and word-to-node alignment.

Annotations are organized by document and by sentence. For each sentence, the annotation contains three parts: the sentence-level graph, the document-level annotation, and the word-to-node alignment. Figure 1 gives an example. Note that alignment is represented as a relation between nodes and a possibly empty set of token-index ranges.

We do not train on the full dirty split, as explained below.

Proceedings of The Seventh International Workshop on Designing Meaning Representations (DMR 2026) @ LREC 2026, pages 155–159

```
# meta-info :: sent_id = u_tree-cs-s2-
root
# :: snt2
Index: 1  2  3     4 5     6     7
 8   9      10
Words: If it rains , Alana won't water
 the plants .

# sentence level graph:
(s2w / water-01
    :ARG0 (s2p / person
        :name (s2n / name :op1 "Alana"))
    :ARG1 (s2p2 / plant
        :refer-number plural)
    :condition (s2r / rain-01
        :aspect process)
    :aspect performance)

# alignment:
s2w: 7-7
s2p: 5-5
s2n: 0-0
s2p2: 9-9
s2r: 3-3

# document level annotation:
(s2s0 / sentence
    :temporal ((document-creation-
time :after s2w)
               (document-creation-
time :after s2r))
    :modal ((root :modal author)
            (author :full-
affirmative s2r)
            (author :full-
negative s2w)))
```

Figure 1: Example of annotation for a sentence.

2. System architecture

2.1. Overview

We fine-tune Qwen3 4B (Yang et al., 2025) with LoRA (Hu et al., 2022). The model predicts the intra-sentential fragment of UMR, namely the sentence-level graph and the word-to-node alignment, as well as part of the document-level annotation, mainly modal relations.

Our training pipeline consists of three stages:

1. training sentence-graph generation on a capped subset of the dirty split;
2. continuing training on the clean split while adding document-level annotations;
3. further fine-tuning on the clean split for alignment generation.

For graph prediction, the model is trained to produce a PENMAN-style representation of the graph without tab characters. Appendix A gives input and output templates for each of the different stages. We include both the language of the input sentence and its id, as node identifiers are always of the form `s{sentence id}`.

For alignment prediction, we do not exactly use the same format as the original data. Instead of generating node-to-word alignment, we generate word-to-node alignment. This yields a more natural generation order and makes it possible to leave words unaligned explicitly when needed.

All experiments are run on a single A100 GPU with 40GB of memory. Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER, several universities, and other organizations.

2.2. Data preparation

As mentioned above, the training data is highly unbalanced. In particular, the dirty split contains a very large number of Czech examples and much fewer samples for other languages. We found that training on the full dirty split leads to prohibitive training times for limited gains in performance. We therefore cap the number of dirty examples per language at 2000 and sample accordingly.

We also reverse the alignment direction and generate alignments in a word-to-node format. Concretely, alignments are represented line by line as `word:node` or `word:` when no node is aligned.

We normalize sentences by trimming each tab character with a space and collapsing repeated spaces into a single space. We also normalize graphs by removing all tab characters. The data includes morphological and syntactic annotations for Navajo and Arapaho, but we do not include them in the model input, in order to keep a consistent format across languages. Note that we don't use any special linearization technic for the graphs that was done for AMR in variants of SPRING (Bevilacqua et al., 2021). We simply use a PENMAN-style linearization without tab characters, which is sufficient to obtain well-formed outputs.

2.3. Training

We use LoRA (Hu et al., 2022) to fine-tune Qwen3 4B (Yang et al., 2025). We apply LoRA to both attention and feed-forward projections. We use different hyperparameters for the dirty and clean stages; the alignment stage uses the same hyperparameters as the clean stage. Table 2 summarizes the training setup.

We train the model with the HuggingFace implementation of LoRA and the `transformers` li-

brary. We use the AdamW optimizer and a constant learning-rate schedule with warmup. In preliminary experiments, a small number of warmup steps gave slightly better performance. Because the per-device batch size is small, we use gradient accumulation.

We reset the optimizer and scheduler states between stages, while initializing each stage from the model obtained at the previous one. For this reason, we also keep warmup in the clean stage, even though it starts from a previously trained checkpoint.

We experimented extending the dirty split with alignment annotations using on a simple Levenshtein-based heuristic, but this did not yield improvements and was not included in the final training pipeline.

3. Generation

We generate annotations in two steps. First, we generate the sentence-level graph together with the document-level annotation using the model saved at the end of the clean stage. Second, we generate word-to-node alignments using the model saved at the end of the alignment stage.

We decode the test set with beam search using a beam size of 4. The generation parameters are given in Table 3. For very long sentences, generation can become unstable: the model sometimes falls into loops and keeps introducing new nodes. When this happens, we restart the generation with the same configuration and keep the first valid output obtained after post-processing.

4. Post-processing

Because the official evaluator requires a specific output format, we apply several post-processing steps to the generated annotations. First, we parse the generated PENMAN-style graph and check whether it is well formed. When parsing fails, we apply simple repair heuristics : inserting missing parentheses or removing unmatched ones if they can be deteced. We then convert the graph to the format expected by the official evaluator.

We also convert alignments from our word-to-node format back to the official format and add empty alignments for nodes that are not aligned to any word. Alignment entries referring to non-existing nodes are removed. Finally, when the document-level annotation is missing, we insert an empty annotation block.

If the output still cannot be parsed after these steps, we fall back to an empty sentence-level graph and remove the corresponding alignment annotation.

5. Results

We observe a general agreement with the expected format, the main errors being related to the introduction of cycles in the generated graphs, visible by the fact that some nodes dominate themselves in the generated PENMAN.

Apart from this issue, the model generally produces well-formed annotations strictly following the expected format. As argued in (Xin et al., 2024), the ability to produce well-formed outputs is a key advantage of LoRA fine-tuning, and must be distinguished from the ability to produce semantically correct outputs. In our case, the model is able to learn to produce well-formed annotations, but performance remains limited in terms of semantic correctness, and especially for navajo and arapaho. Table 4 shows the official $\mathrm{ju{:}mætʃ}$ scores. The results are not uniform across languages, with very low scores for Navajo and Arapaho, and better results for Chinese, Czech, and Italian. The average score across languages is 0.1943, with lowest scores 0.1115 for Arapaho and highest score 0.2652 for Czech.

A qualitative analysis of the outputs shows that, for Navajo and Arapaho, the model often produces very generic and sparse annotations, as illustrated in Figure 2. This is likely due to the very small number of training examples for these languages and to their limited representation in the model pretraining data.

By contrast, Italian, which is absent from the fine-tuning data but likely present in the model pretraining data, obtains better results, as shown in Table 4. This is consistent with the hypothesis that multilingual pretraining enables some transfer to languages that are not present in task-specific fine-tuning, provided that they are sufficiently represented during pretraining. At the same time, the low scores for very low-resource languages indicate that pretraining alone is not enough in the most data-scarce settings.

Conclusion

We presented the Sema system for the DMR 2026 shared task on multilingual parsing from natural language to UMR. Our approach relies on multi-stage LoRA fine-tuning of Qwen3-4B, using capped dirty-data training, continuation on the clean split, a dedicated alignment generation stage, and rule-based post-processing. The results show that this approach is viable across several languages and that it exhibits encouraging cross-lingual transfer to Italian despite its absence in the fine-tuning data. However, performance remains limited for all languages, especially for very low-resource languages such as Navajo and Arapaho.

	dirty	clean / alignment
per-device train batch size	1	1
gradient accumulation steps	32	32
warmup steps	4	3
num train epochs	5	3
learning rate	2e-4	1e-4
weight decay	0.001	0.001
lr scheduler type	constant with warmup	constant with warmup
lora r	16	16
lora alpha	32	32
lora dropout	0	0

Table 2: Training hyperparameters for each stage.

	sentence graph & document-level annotation	alignment
max new tokens	1024	1024
num beams	4	4
temperature	0.7	0.7
top p	0.9	0.9

Table 3: Generation parameters for each step.

```
# :: snt43
Words: Noh ne'nih'iisnoo3woohok
 nouun nehe' hisi' .

# sentence level graph:
(s43h / hii-00
    :actor (s43p / person
        :refer-person 3rd
        :refer-number singular)
    :undergoer (s43p2 / person
        :refer-person 2nd
        :refer-number singular)
    :aspect performance)
```

Figure 2: Example of a generated annotation for a Navajo sentence.

Language	F1
Arapaho	0.1115
Chinese	0.2585
Czech	0.2652
English	0.1919
Italian	0.2137
Latin	0.1918
Navajo	0.1273
Average	0.1943

Table 4: Score for each language and the macro-average reported by the shared task.

6. Bibliographical 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*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12564–12573.

Julia Bonn, Matthew J. Buchholz, Jayeol Chun, Andrew Cowell, William Croft, Lukas Denk, Sijia Ge, Jan Hajič, Kenneth Lai, James H. Martin, Skatje Myers, Alexis Palmer, Martha Palmer, Claire Benet Post, James Pustejovsky, Kristine Stenzel, Haibo Sun, Zdeňka Urešová, Rosa Vallejos, Jens E. L. Van Gysel, Meagan Vigus, Nianwen Xue, and Jin Zhao. 2024. Building a broad infrastructure for uniform meaning representations. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2537–2547, Torino, Italia. ELRA and ICCL.

Jayeol Chun and Nianwen Xue. 2024. Uniform meaning representation parsing as a pipelined

approach. In *Proceedings of TextGraphs-17: Graph-based Methods for Natural Language Processing*, pages 40–52, Bangkok, Thailand. Association for Computational Linguistics.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3.

Emma Markle, Javier Gutierrez Bach, and Shira Wein. 2026. Setup: Sentence-level english-to-uniform meaning representation parser.

Chunlei Xin, Yaojie Lu, Hongyu Lin, Shuheng Zhou, Huijia Zhu, Weiqiang Wang, Zhongyi Liu, Xianpei Han, and Le Sun. 2024. Beyond full fine-tuning: Harnessing the power of LoRA for multi-task instruction tuning. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2307–2317, Torino, Italia. ELRA and ICCL.

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.

Jan Štěpánek, Daniel Zeman, Markéta Lopatková, Federica Gamba, Hana Hledíková, and Nianwen Xue. 2026. First shared task on UMR parsing. In *Proceedings of the Seventh International Workshop on Designing Meaning Representations*, Palma, Spain. ELRA.

A. Prompts and templates

This appendix gives simplified input and output templates for the main stages of our pipeline.

Stage 1: sentence-level graph generation
Input:

```
Parse into the Uniform Mean-
ing Representation the fol-
lowing <language> sen-
tence <num> : " <text> ". You only need to in-
clude the sentence level graph.
```

Output:

```
# sentence level graph:
<sentence_graph>
```

Stage 2: sentence-level graph generation and document-level annotation
Input:

```
Parse into the Uniform Mean-
ing Representation the fol-
lowing <language> sen-
tence <num> : " <text> ". You need to in-
clude sentence level graph and docu-
ment level graph.
```

Output:

```
# sentence level graph:
<sentence_graph>
# document level annotation:
<document_level_annotation>
```

Stage 3: alignment generation
Input:

```
Complete with word to to-
ken alignment the follow-
ing Uniform Meaning Represen-
tation annotation for the sen-
tence " <text> " :  <sentence_graph>
```

Output:

```
# alignment:
<word1> : <node1>
...
<wordN> : <nodeM>
```