PALI at SemEval-2026 Task 3: LoRA Fine-Tuning with Validation for DimABSA

Cheng Chen

oscarhscc@gmail.com

Abstract

We describe the PALI system submitted to SemEval-2026 Task 3 (Dimensional Aspect-Based Sentiment Analysis), which requires predicting valence–arousal (VA) scores and extracting structured sentiment tuples across multiple languages. Our final system centers on LoRA fine-tuning of Qwen3-32B using Llama-Factory, together with data conversion/cleaning, multilingual data-mixing strategies, and inference-time validation and repair. We additionally explored retrieval-based few-shot prompting with BGE-M3, but found it less effective for learning consistent VA scoring preferences. On Track A, our final system uses per-language LoRA adapters that mix all subtasks per language for a better trade-off between performance and efficiency. On the official test set, we achieve average per-language scores of 1.2071 RMSE_VA for Subtask 1 and 0.5641/0.4905 cF1 for Subtask 2/3. On the development set, we find that per-language-per-task adapters further improve extraction cF1 but are less attractive in terms of training and deployment cost. For Track B, we report results for VA prediction on five languages and two domains.

1 Introduction

SemEval-2026 Task 3 focuses on *dimensional* sentiment analysis, where sentiment is represented as continuous *valence* and *arousal* values rather than discrete polarity labels. The shared task includes three subtasks: (i) predicting VA scores for given aspect terms (Subtask 1), (ii) extracting (Aspect, Opinion, VA) triplets (Subtask 2), and (iii) extracting (Aspect, Category, Opinion, VA) quadruplets with categories constrained to a predefined ontology (Subtask 3). The task is further complicated by multilingual data, strict output formatting, and the need to keep extracted spans faithful to the input text. Track A covers English, Chinese, Japanese, Russian, Tatar, and Ukrainian, while Track B covers German, English, Nigerian Pidgin, Swahili, and Chinese. [Lee et al., 2026, Becker et al., 2026]

We follow the official task overview paper [Yu et al., 2026] and cite the released dataset papers for Track A and Track B [Lee et al., 2026, Becker et al., 2026].

Our work is also related to prior research on aspect-based sentiment analysis and affect prediction. Traditional ABSA benchmarks and shared tasks focus primarily on categorical sentiment labels and structured extraction [Pontiki et al., 2014, 2016, Hua et al., 2024], whereas valence–arousal prediction has been studied in lexicon induction and tweet-level affect analysis [Yu et al., 2015, Mohammad et al., 2018]. DimABSA brings these two lines of work together by requiring both structured aspect-level extraction and continuous affect prediction.

We submit systems for Track A and Track B. Our main contributions are: (1) a data conversion and cleaning pipeline that standardizes the official `*_alltasks.jsonl` files into subtask-specific formats and removes problematic training instances; (2) a systematic comparison of LoRA fine-tuning strategies with different language/task mixing granularities; and (3) an inference-time validation and repair pipeline that improves robustness in languages where empty or ungrounded spans are more frequent.

Following SemEval requirements, we emphasize system description and analysis, and focus on replicability.

2 System Overview

Figure 1 summarizes our pipeline. We first preprocess the official data into subtask-specific files and language splits. We then train LoRA adapters on top of Qwen3-32B using instruction-style supervision for structured JSON outputs. At inference time, we apply strict validators for JSON schema,

Proceedings of the The 20th International Workshop on Semantic Evaluation (2026), pages 641–649

Data → conversion & cleaning → (language/task) splits → LoRA fine-tuning (Qwen3-32B) → inference → validation & repair → final predictions

Figure 1: High-level pipeline of the PALI system.

numerical range constraints, and text-grounding checks; when a sample fails validation, we trigger re-prediction with fallback models.

Modeling perspective. All three subtasks can be viewed as a unified *structured prediction* problem under a shared VA regression scale. Subtask 1 reduces to calibrated regression for a fixed list of aspects, whereas Subtask 2/3 additionally require faithful span extraction (and in Subtask 3, ontology-constrained categorization). In our implementation, we express all subtasks as instruction-following generation with strict JSON schemas, enabling a single fine-tuning interface and consistent validators across tasks.

3 Data Processing

3.1 Data conversion

The official release provides files with the suffix `*_alltasks.jsonl`. We convert each record into subtask-specific instances for Subtask 1/2/3 with the required input/output fields and JSON schemas. To enable controlled training strategies, we additionally bucket the converted data by **language** and **subtask**.

Schema normalization. Besides splitting by subtask, we normalize field names and formatting constraints so that each instance can be expressed as an instruction with (i) an input text, (ii) optional domain metadata, and (iii) a required output schema. This normalization simplifies multi-task mixing and reduces training noise caused by inconsistent serialization.

3.2 Data cleaning

We apply the following cleaning rules to the **training** split:

1. Remove instances whose aspect field is the string `"NULL"`.
2. Remove instances with duplicated aspect terms.[1]

[1]We treat duplicates as repeated aspect strings within the same input instance.

We keep the development and test sets unchanged.

Rationale. We observed that `"NULL"` aspects and duplicated aspects can introduce contradictory supervision: the former provides no opinion target for dimensional scoring, while the latter can encourage inconsistent VA assignments for identical strings within a single instance. Filtering these cases improves training stability and reduces avoidable formatting failures in generation.

4 Methods

4.1 Task constraints and output formats

All subtasks require strict adherence to JSON output formats and value constraints. For Subtask 1, aspect terms must be used *exactly* as provided in the input (order- and string-preserving), and VA scores must be real numbers in $[1.00, 9.00]$ rounded to two decimals. For Subtask 2 and Subtask 3, extracted aspect and opinion terms must be verbatim spans from the input text, and Subtask 3 categories must be selected from the predefined ontology.

Why formatting matters. In preliminary experiments we found that a non-trivial fraction of errors are not semantic but *structural* (e.g., invalid JSON, missing keys, empty lists, or out-of-range values), especially in low-resource languages. Therefore, we treat format correctness as a first-class objective via explicit schema instructions and inference-time validators.

4.2 Retrieval-based few-shot prompting (exploratory)

We explored a retrieval-augmented few-shot approach for Track A. For each language and subtask, we embed training instances using BGE-M3 [Chen et al., 2024] and build a vector index. During development inference, we retrieve the top-$k = 10$ most similar examples and concatenate them into the prompt before querying different API models (Kimi-K2, Gemini-3.0-Flash, and Grok4.1). Overall, we found that few-shot prompting did not reliably capture the global scoring preferences needed for stable VA regression, and therefore we focused on fine-tuning.

Discussion. We hypothesize that this limitation stems from two factors: (i) VA prediction requires *global calibration* to the annotation scale (e.g., the typical distance from neutrality), and (ii) retrieval can over-emphasize local lexical similarity while

missing domain-level or language-specific scoring conventions.

4.3 Fine-tuning with LoRA

4.3.1 Base model selection

We initially attempted full-parameter fine-tuning with Qwen3-4B-Instruct-2507 on H800 GPUs, but the results were not satisfactory. We then switched to parameter-efficient fine-tuning with LoRA [Hu et al., 2021] and adopted Qwen3-32B [Yang et al., 2025] as our primary dense backbone, as we observed better scaling behavior with larger parameter counts. We also considered Qwen3-30B-A3B-Instruct-2507 (MoE), but abandoned it due to low fine-tuning efficiency and high cost.

Motivation for LoRA. LoRA [Hu et al., 2021] provides a practical way to adapt a large backbone with limited compute, while also enabling modular routing (e.g., selecting different adapters by language). This modularity is particularly useful in multilingual settings where a single mixed model may underfit low-resource languages.

4.3.2 Training framework and objective

We fine-tune with Llama-Factory [Zheng et al., 2024] using supervised instruction tuning [Chung et al., 2022]. Each training instance is formatted as an instruction to produce strictly valid JSON output for the corresponding subtask. **Hyperparameters.** We use a maximum sequence length of 4096 tokens, LoRA rank $r = 8$, and train for 1 epoch. We fine-tune with bf16 enabled, a learning rate of 1.0×10^{-4}, and batch size 1. For multi-task training, we mix Subtask 1/2/3 instances with an equal sampling ratio. Given the limited submission-time budget, we did not run a broad sweep over LoRA ranks or epoch counts; instead, we fixed a lightweight configuration and compared mixing strategies under the same setting.

Prompting template (high level). We use a consistent instruction format that includes (i) a short task description, (ii) the input text (and domain if provided), and (iii) a minimal JSON schema that the model must follow. For Subtask 1, we explicitly restate that aspects must match the provided list exactly; for Subtask 2/3, we emphasize verbatim span extraction and prohibit hallucinated aspects not present in the text. Full prompt specifications used for each subtask are provided in Appendix A.

4.3.3 Language/task data mixing strategies (Track A)

We compare three strategies:

1. **S1: All-in-one.** Mix all languages and all subtasks (Subtask 1/2/3) with global shuffling and train a single LoRA adapter.

2. **S2: Per-language.** For each language, mix Subtask 1/2/3 and train one LoRA adapter (6 adapters in total).

3. **S3: Per-language-per-task.** Train one LoRA adapter for each (language, subtask) pair (18 adapters in total).

At inference time, we route each test instance to the corresponding adapter (S2/S3) based on its language and task.

Final submission choice. Although S3 can yield stronger extraction scores on the development set (Table 9), it requires maintaining 18 adapters (one per language and subtask), which increases training time, storage, and deployment complexity. Given the overall time–cost–performance trade-off, we select S2 (per-language adapters, 6 in total) as our final Track A system.

4.4 Inference-time validation and post-processing

Our post-processing aims to prevent invalid submissions and to reduce errors caused by empty or ungrounded extractions. We apply:

1. **Format validation:** JSON parseability, required keys, numeric range $[1.00, 9.00]$, and two-decimal rounding; for Subtask 1 we additionally check that the output aspects match the input aspects exactly in count and order.

2. **Empty-output repair:** if the predicted aspect list is empty (e.g., `[]`), we trigger re-prediction with a stronger fallback model using a consistent few-shot prompt. We first call Claude 4.5 Sonnet; if still empty, we call ChatGPT-5.2 as a third attempt.

3. **Text-grounding check:** if a predicted aspect is not found as a substring of the input text (frequent in Russian, Tatar, and Ukrainian), we trigger the same re-prediction cascade as above.

Language	Subtask 1	Subtask 2	Subtask 3
ENG	0.90	1.75	3.35
JPN	21.31	15.38	16.50
RUS	16.70	9.21	14.13
TAT	12.87	14.44	18.73
UKR	11.66	6.67	11.75
ZHO	0.49	0.65	1.55

Table 1: Estimated trigger rates (%) for the inference-time repair stage on Track A, computed as 1− first-pass validation rate from the recorded pass counts.

Subtask	Repair success rate
Subtask 1	$> 99\%$
Subtask 2	$> 99\%$
Subtask 3	$> 99\%$

Table 2: Observed repair effectiveness. The small number of remaining unresolved cases were corrected manually before final submission.

Practical impact. These validators reduce submission failures and mitigate common generation issues, such as empty lists in Japanese and ungrounded spans in Cyrillic scripts, making the pipeline more robust for multilingual evaluation. Using first-pass validator counts recorded during inference, we estimate the trigger rate as one minus the proportion of instances that pass validation on the first attempt. Table 1 shows that trigger rates are very low for English and Chinese, but substantially higher for Japanese and for some lower-resource or morphologically rich languages. Micro-averaged over languages, the estimated trigger rates are 8.44%, 5.41%, and 7.64% for Subtask 1/2/3, respectively.

Repair effectiveness. Across subtasks, the repair stage resolves more than 99% of triggered cases. In practice, the fallback API stage acts as a targeted safety net rather than the primary predictor: most English and Chinese instances pass validation on the first attempt, while the repair cascade is mainly needed for languages where empty outputs or ungrounded spans are more common.

VA prediction as constrained generation. We model VA prediction as text generation rather than explicit regression, following the now-common practice of casting diverse prediction problems as instruction-following generation [Chung et al., 2022]. This design lets us use a single instruction-tuning interface for all three subtasks and enforce a shared JSON schema at decoding time. Numerical constraints are then handled by the validator, which checks value range and rounding. A limitation of this choice is that the model is optimized with token-level supervision rather than a regression loss directly aligned with RMSE; exploring hybrid regression-generation objectives is an important direction for future work.

5 Experiments

5.1 Experimental setup

We train and evaluate systems for Track A and Track B. All fine-tuning experiments were run on H800 GPUs. We report the official shared-task metrics for each subtask: RMSE_VA for Subtask 1 and cF1 for Subtask 2/3. Unless stated otherwise, reported development numbers are macro-averaged across languages (Table 9). Due to time constraints, we report single runs for each configuration in this camera-ready version.

Macro-averaging. When reporting "average" development performance for ablations, we use unweighted macro-averages over the language-level `AVERAGE` rows (rather than weighting by dataset size) to avoid letting high-resource languages dominate the comparison.

5.2 Development results for the mixed-language LoRA baseline

Table 3–Table 5 report our development-set results for a single LoRA adapter trained by mixing all languages and all subtasks (S1 in Section 4.3.3). We include per-language, per-domain scores and language averages.

5.3 Main results

Table 6–Table 8 report our final Track A results on the official test set. We include per-language, per-domain scores and language averages.

5.4 Ablation: data mixing and routing (Track A)

Table 9 compares S1–S3 on the development set. S2 improves Subtask 1 RMSE_VA over S1 (1.1086 vs. 1.1928), while S3 yields the best extraction performance on Subtask 2/3 (0.6092/0.5486). We therefore choose S2 for the final submission because it offers a better efficiency–performance trade-off than S3 while remaining stronger than S1 on two of the three reported metrics.

Language	Domain	RMSE_VA ↓
JPN	finance	1.0506
JPN	hotel	0.8831
JPN	AVERAGE	0.9668
RUS	restaurant	1.2949
RUS	AVERAGE	1.2949
TAT	restaurant	1.6653
TAT	AVERAGE	1.6653
UKR	restaurant	1.4389
UKR	AVERAGE	1.4389
ZHO	finance	0.7266
ZHO	laptop	0.6837
ZHO	restaurant	0.7084
ZHO	AVERAGE	0.7062
ENG	laptop	1.1137
ENG	restaurant	1.0530
ENG	AVERAGE	1.0834

Table 3: Subtask 1 (DIMASR): development-set RMSE_VA for the mixed-language LoRA baseline.

Language	Domain	cF1 ↑
ENG	laptop	0.5558
ENG	restaurant	0.6995
ENG	AVERAGE	0.6276
ZHO	laptop	0.4939
ZHO	restaurant	0.6485
ZHO	AVERAGE	0.5712
JPN	hotel	0.5469
JPN	AVERAGE	0.5469
RUS	restaurant	0.5251
RUS	AVERAGE	0.5251
TAT	restaurant	0.4811
TAT	AVERAGE	0.4811
UKR	restaurant	0.4678
UKR	AVERAGE	0.4678

Table 4: Subtask 2 (DIMASTE): development-set cF1 for the mixed-language LoRA baseline.

Language	Domain	cF1 ↑
RUS	restaurant	0.5267
RUS	AVERAGE	0.5267
ZHO	laptop	0.4040
ZHO	restaurant	0.5859
ZHO	AVERAGE	0.4950
ENG	laptop	0.2874
ENG	restaurant	0.6599
ENG	AVERAGE	0.4737
UKR	restaurant	0.4372
UKR	AVERAGE	0.4372
TAT	restaurant	0.4117
TAT	AVERAGE	0.4117
JPN	hotel	0.4095
JPN	AVERAGE	0.4095

Table 5: Subtask 3 (DIMASQP): development-set cF1 for the mixed-language LoRA baseline.

Language	Domain	RMSE_VA ↓
ENG	laptop	1.3612
ENG	restaurant	1.2866
ENG	AVERAGE	1.3239
JPN	finance	0.7532
JPN	hotel	0.6237
JPN	AVERAGE	0.6884
RUS	restaurant	1.3642
RUS	AVERAGE	1.3642
TAT	restaurant	1.7121
TAT	AVERAGE	1.7121
UKR	restaurant	1.4030
UKR	AVERAGE	1.4030
ZHO	finance	0.6042
ZHO	laptop	0.6681
ZHO	restaurant	0.9805
ZHO	AVERAGE	0.7510

Table 6: Track A test results for Subtask 1 (DIMASR): RMSE_VA.

5.5 Track B test results

Track B provides test predictions only for Subtask 1 (VA prediction). Table 10 summarizes our final Track B results.

5.6 Analysis and discussion

We qualitatively observed that retrieval-based few-shot prompting is less stable for VA regression than LoRA fine-tuning, likely because VA prediction requires global calibration to the annotation scale.

Mixing strategy comparison. Table 9 highlights a clear trade-off between simplicity, performance, and operational cost. For Subtask 1, the per-language strategy (S2) achieves the best dev RMSE_VA (1.1086), improving over the all-in-one baseline (S1, 1.1928). In contrast, for extraction (Subtask 2/3), the per-language-per-task strategy (S3) substantially improves cF1 (0.6092/0.5486), suggesting that extraction benefits from task-specialized adapters. However, S3 requires 18 LoRA adapters, which increases both training and inference overhead. We therefore adopt S2 as our final submission due to its better efficiency while remaining competitive on all subtasks.

Per-language variation and domain effects. On Track A test, we observe that Japanese (JPN) and Chinese (ZHO) are consistently easier for Subtask 1

Language	Domain	cF1 ↑
ENG	laptop	0.6242
ENG	restaurant	0.6928
ENG	AVERAGE	0.6585
RUS	restaurant	0.5724
RUS	AVERAGE	0.5724
UKR	restaurant	0.5671
UKR	AVERAGE	0.5671
JPN	hotel	0.5566
JPN	AVERAGE	0.5566
ZHO	laptop	0.5308
ZHO	restaurant	0.5634
ZHO	AVERAGE	0.5471
TAT	restaurant	0.4828
TAT	AVERAGE	0.4828

Table 7: Track A test results for Subtask 2 (DIMASTE): cF1.

(JPN avg 0.6884, ZHO avg 0.7510) than the low-resource and morphologically rich languages (TAT 1.7121, UKR 1.4030). For extraction, English and restaurant-domain instances tend to yield higher cF1 than laptop/hotel in some cases (e.g., ENG laptop vs. restaurant in Subtask 3: 0.3793 vs. 0.6395), suggesting that domain-specific phrasing and annotation density can strongly affect extraction quality.

Why S2 can outperform S1 for VA prediction. The improvement of S2 over S1 on Subtask 1 dev RMSE_VA (1.1086 vs. 1.1928) suggests that language-specific adapters can better learn distributional properties of VA labels (e.g., language-dependent intensity markers and hedging) without interference from other languages during optimization. This is consistent with the observation that low-resource languages (e.g., Tatar) remain challenging even after adaptation, but benefit from not being overshadowed by larger languages in a fully mixed setup.

Why S3 helps extraction but is not chosen. S3 improves cF1 substantially (dev Subtask 2/3: 0.6092/0.5486), plausibly because task specialization reduces negative transfer between regression-style outputs (Subtask 1) and span-centric extraction outputs (Subtask 2/3). Nevertheless, maintaining 18 adapters increases both engineering burden and inference latency. Given that our final evaluation prioritizes an end-to-end pipeline that is stable and cost-efficient, we choose S2 for final submission.

Language	Domain	cF1 ↑
RUS	restaurant	0.5496
RUS	AVERAGE	0.5496
UKR	restaurant	0.5307
UKR	AVERAGE	0.5307
ENG	laptop	0.3793
ENG	restaurant	0.6395
ENG	AVERAGE	0.5094
ZHO	laptop	0.4319
ZHO	restaurant	0.5357
ZHO	AVERAGE	0.4838
TAT	restaurant	0.4443
TAT	AVERAGE	0.4443
JPN	hotel	0.4252
JPN	AVERAGE	0.4252

Table 8: Track A test results for Subtask 3 (DIMASQP): cF1.

Strategy	Subtask 1	Subtask 2	Subtask 3
S1: All-in-one	1.1928	0.5367	0.4592
S2: Per-language	1.1086	0.5289	0.4737
S3: Per-language-per-task	1.2033	0.6092	0.5486

Table 9: Ablation on Track A development set for different mixing strategies. For Subtask 1 we report macro-average RMSE_VA (lower is better); for Subtask 2/3 we report macro-average cF1 (higher is better) across languages.

Untuned backbone behavior. We do not consider an untuned Qwen3-32B model a competitive baseline for the final system. In preliminary zero-shot use, it showed two major weaknesses: first, instruction-following for long multilingual JSON outputs was less reliable, which directly hurt format validity; second, without task-specific label supervision, it lacked the calibration needed for VA prediction and the structured extraction ability required by Subtask 2/3. This observation further motivates our combination of supervised fine-tuning and lightweight repair.

Limitations of current analysis. Although we now report trigger rates and aggregate repair effectiveness, we did not preserve a separate official no-repair run that would allow a clean metric-level comparison between raw backbone outputs and repaired outputs on the test set.

Language	Domain	RMSE_VA ↓
DEU	politics	1.5688
DEU	AVERAGE	1.5688
ENG	environmental_protection	1.8048
ENG	AVERAGE	1.8048
PCM	politics	1.4078
PCM	AVERAGE	1.4078
SWA	politics	2.4544
SWA	AVERAGE	2.4544
ZHO	environmental_protection	0.7047
ZHO	AVERAGE	0.7047

Table 10: Track B test results for Subtask 1 (VA prediction): RMSE_VA.

6 Limitations and Ethics Statement

Our system relies on external API models during post-processing, which introduces additional cost, latency, and reproducibility risks if model versions change. We did not fully explore more compute-intensive fine-tuning of MoE backbones due to efficiency constraints. We follow the shared task's data usage terms and focus on producing faithful, text-grounded extractions without injecting external knowledge. However, because the fallback stage sends task inputs to third-party APIs, the current pipeline may be unsuitable for privacy-sensitive deployment settings even when it is acceptable for shared-task evaluation on public benchmark data.

Replicability note. While our data conversion and cleaning rules, along with our overall training strategies, are described in this paper, some implementation details (exact prompts, full hyperparameters, and fallback model configurations) are not fully specified because of time constraints and the use of closed-source APIs in post-processing. We encourage future work to replace the fallback stage with constrained decoding or schema-guided generation to reduce external dependencies.

7 Conclusion

We presented the PALI system for SemEval-2026 Task 3. Our best-performing approach uses LoRA fine-tuning of Qwen3-32B with carefully designed language/task mixing strategies and a robust inference-time validation and repair pipeline. In future work, we plan to reduce reliance on external APIs and explore constrained decoding to enforce schema and span-faithfulness more directly.

Acknowledgments

We thank the organizers of SemEval-2026 Task 3 for releasing the dataset and evaluation framework.

A Prompts

A.1 Subtask 1 prompt (DIMASR: VA prediction)

```
## Task Description
- Input: A text (e.g., a review or
↪  headline) and one or more predefined
↪  aspect terms.
- Output: A VA score pair for each given
↪  aspect term.
- Key Rules:
  - Use aspect terms exactly as provided
  ↪  (case-sensitive, same order), with
  ↪  no
    modifications/additions/deletions.
    ↪  Output must match input aspects
    ↪  exactly.
  - Predict VA scores based solely on
  ↪  the text's cues; avoid external
  ↪  knowledge.
  - Consider the domain if specified
  ↪  (Restaurant, Laptop, Hotel,
  ↪  Finance).
  - Support multilingual texts.

## Output Format
Output JSON with key 'Aspect_VA'
↪  containing a list of objects:
- 'aspect': the exact aspect term from
↪  input (string)
- 'valence': float in [1.00, 9.00], two
↪  decimals
- 'arousal': float in [1.00, 9.00], two
↪  decimals
```

A.2 Subtask 2 prompt (DIMASTE: triplet extraction)

```
## Task Description
- Input: A single text (e.g., a news
↪  headline).
- Output: All identified (A, O, VA)
↪  triplets.
- Key Rules:
  - Extract Aspect Terms verbatim from
  ↪  the text; do not modify or infer.
  - Identify all valid triplets; do not
  ↪  omit any or invent new ones.
  - Base extractions on explicit
  ↪  content.

## Output Format
Output JSON with key 'Triplet'
↪  containing a list of objects:
- 'Aspect': exact aspect term from text
- 'Opinion': associated opinion term
- 'VA': 'V#A' with V and A floats in
↪  [1.00, 9.00], two decimals
```

A.3 Subtask 3 prompt (DIMASQP: quadruplet extraction)

```
## Task Description
- Input: A single text (e.g., a news
↪  headline).
```

```
- Output: All identified (A, C, O, VA)
↪  quadruplets.
- Key Rules:
  - Extract Aspect Terms verbatim from
  ↪  the text; do not modify or infer.
  - Assign Aspect Categories strictly
  ↪  from the predefined ontology
  ↪  (Entity#Attribute).
  - Identify all valid quadruplets; do
  ↪  not omit any or invent new ones.
  - Base extractions on explicit
  ↪  content.

## Output Format
Output JSON with key 'Quadruplet'
↪  containing a list of objects:
- 'Aspect': exact aspect term from text
- 'Category': predefined category
↪  (Entity#Attribute, UPPERCASE)
- 'Opinion': associated opinion term
- 'VA': 'V#A' with V and A floats in
↪  [1.00, 9.00], two decimals
```

References

Jonas Becker, Liang-Chih Yu, Shamsuddeen Hassan Muhammad, Jan Philip Wahle, Terry Ruas, Idris Abdulmumin, Lung-Hao Lee, Nelson Odhiambo, Lilian Wanzare, Wen-Ni Liu, Tzu-Mi Lin, Zhe-Yu Xu, Ying-Lung Lin, Jin Wang, Maryam Ibrahim Mukhtar, Bela Gipp, and Saif M. Mohammad. Dimstance: Multilingual datasets for dimensional stance analysis, 2026. URL https://arxiv.org/abs/2601.21483.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *CoRR*, abs/2402.03216, 2024.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, et al. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416, 2022.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685, 2021.

Yan Cathy Hua, Paul Denny, Jorg Wicker, and Katerina Taskova. A systematic review of aspect-based sentiment analysis: Domains, methods, and trends. *Artificial Intelligence Review*, 57:296, 2024.

Lung-Hao Lee, Liang-Chih Yu, Natalia Loukashevich, Ilseyar Alimova, Alexander Panchenko, Tzu-Mi Lin, Zhe-Yu Xu, Jian-Yu Zhou, Guangmin Zheng, Jin Wang, Sharanya Awasthi, Jonas Becker, Jan Philip Wahle, Terry Ruas, Shamsuddeen Hassan Muhammad, and Saif M. Mohammad. Dimabsa: Building multilingual and multidomain datasets for dimensional aspect-based sentiment analysis, 2026. URL https://arxiv.org/abs/2601.23022.

Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 1–17, 2018.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, 2014.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphee De Clercq, Veronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud Maria Jimenez-Zafra, and Gulsen Eryigit. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, 2016.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025.

Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xue-jie Zhang. Predicting valence-arousal ratings of words using a weighted graph method. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 788–793, 2015.

Liang-Chih Yu, Jonas Becker, Shamsuddeen Hassan Muhammad, Idris Abdulmumin, Lung-Hao Lee, Ying-Lung Lin, Jin Wang, Jan Philip Wahle, Terry Ruas, Alexander Panchenko, Ilseyar Alimova, Kai-Wei Chang, Lilian Wanzare, Nelson Odhiambo, Bela Gipp, and Saif M. Mohammad. SemEval-2026 task 3: Dimensional aspect-based sentiment analysis (DimABSA). In *Proceedings of the 20th International Workshop on Semantic Evaluation (SemEval-2026)*. Association for Computational Linguistics, June 2026.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *CoRR*, abs/2403.13372, 2024.