Rubric-as-Experts: Case-Specific MQM Rubrics for Translation Quality Evaluation

Anonymous ACL submission

Abstract

Large language models (LLMs) have shown strong potential in fine-grained translation quality evaluation (QE), yet existing MQM-based approaches typically rely on fixed rubric configurations shared across all translation samples. However, translation instances often differ substantially in error complexity, ambiguity, and required evaluation granularity, making static rubric allocation suboptimal for span-level error detection. We find that larger MQM subtype spaces improve error coverage but also introduce more false positives, while different translation instances prefer different rubric granularities, suggesting that evaluation spaces should be allocated dynamically for each case. Motivated by these observations, we propose a case-specific dynamic rubric framework that adaptively constructs MQM evaluation spaces for individual translation instances. Unlike fully free-form rubric generation methods, our framework remains grounded in the predefined MQM taxonomy while dynamically selecting suitable subtype spaces and evaluation granularity for different cases. Experiments on WMT span-level QE benchmarks across multiple model scales demonstrate that the proposed framework consistently improves MCC and produces cleaner span-level error localization compared with static rubric settings. Our results suggest that combining structured MQM rubrics with case-specific adaptive allocation is an effective strategy for fine-grained LLM-based translation evaluation.

1 Introduction

Large language models (LLMs) have recently demonstrated strong performance in machine translation generation and multilingual reasoning tasks (Brown et al., 2020; Achiam et al., 2023; Yang et al., 2025). In comparison, their advantages in fine-grained translation quality evaluation (QE), particularly span-level error localization, remain less significant than their strong performance gains in translation generation, with existing QE systems such as COMETKiwi (Rei et al., 2022) still remaining highly competitive. Unlike sentence-level evaluation, span-level QE requires models to identify precise error spans under structured evaluation criteria such as MQM (Multidimensional Quality Metrics) (Lommel et al., 2014; Freitag et al., 2021). Recent studies suggest that rubric-guided prompting can improve the consistency and controllability of LLM-based evaluation (He et al., 2024). However, most existing approaches rely on static evaluation settings in which all translation samples share the same rubric granularity and evaluation search space.

In practice, translation quality often admits multiple acceptable interpretations, and the boundary between acceptable variation and genuine translation error can be ambiguous. As a result, span-level QE requires not only error detection ability, but also appropriate control over evaluation strictness and exploration scope. A restricted evaluation search space may reduce error coverage, while an overly broad search space can introduce redundant exploration and excessive false positive predictions. These observations suggest that evaluation search space mismatch may be an important factor affecting span-level QE behavior.

Existing approaches generally follow two directions. One direction uses direct free-form rubric evaluation, where LLMs first generate evaluation rubrics dynamically and then perform error evaluation based on the generated criteria (Chang et al., 2024). Although flexible, such methods often produce unstable evaluation scope, inconsistent coverage, and less controllable reasoning behavior across different cases. Another common direction adopts MQM-based fixed rubric prompting, where all translation samples share the same predefined evaluation granularity (Kim, 2025). While this im-

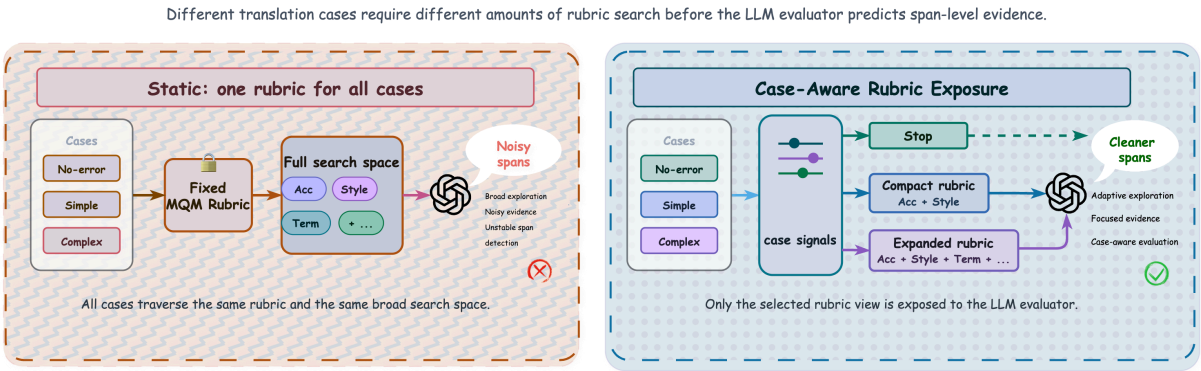

Figure 1: Illustration of static versus case-aware rubric exposure for span-level QE. Static rubric settings expose all translation cases to the same broad MQM search space, which may introduce noisy span predictions and redundant exploration. In contrast, the proposed framework dynamically allocates rubric granularity according to instance-level signals, enabling more focused evaluation behavior and cleaner span-level evidence.

proves evaluation consistency, fixed rubric scope may not adapt well to heterogeneous translation instances. In our preliminary experiments, we further observe that aggressively expanding MQM subtype scope substantially improves recall, but also introduces considerable exploration redundancy and additional false positive predictions. More importantly, different translation samples often exhibit different granularity preferences, suggesting that effective evaluation scope may be highly instance-dependent.

Based on these observations, we propose a case-specific dynamic rubric framework for span-level QE. The proposed framework dynamically allocates MQM evaluation granularity and rubric scope according to individual translation instances. Experiments on span-level QE benchmarks show that dynamic rubric allocation achieves stronger MCC performance and produces cleaner span-level error localization compared with static evaluation settings.

The contributions of this work are summarized as follows:

- We present a systematic analysis of how MQM rubric granularity influences LLM-based span-level QE, showing that fine-grained subtype expansion affects model exploration behavior and exhibits notable instance-level granularity differences.
- We propose a case-specific dynamic rubric framework that adaptively allocates MQM evaluation granularity and evaluation search space according to translation instance characteristics.
- Experimental results show that dynamic rubric allocation improves MCC performance and produces cleaner error localization for span-level QE.

2 Related Work

LLM-based Translation Quality Evaluation Translation quality evaluation (QE) has gradually evolved from sentence-level quality scoring toward fine-grained token- and span-level error detection. Instead of only predicting an overall quality score, recent QE systems aim to identify concrete error spans. MQM provides a structured taxonomy for fine-grained translation evaluation, defining hierarchical error categories such as accuracy, fluency, terminology, and style (Lommel et al., 2014). Building on this framework, recent LLM-based QE methods show that large language models can perform MQM-style analytic evaluation through prompting (Kocmi and Federmann, 2023). These studies suggest that LLMs already possess relatively strong fine-grained error perception ability. However, existing methods typically rely on fixed prompts or static rubric configurations shared across all samples. As a result, the model's error exploration behavior is strongly constrained by a predefined evaluation space, even though different translation cases may require substantially different levels of error exploration and rubric granularity.

Rubric-conditioned Evaluation Rubric-based and criterion-conditioned evaluation have become

increasingly important in LLM-as-a-judge research (Liu et al., 2023; Anghel et al., 2025). By explicitly providing evaluation criteria in natural language, these methods guide the model' s reasoning focus and evaluation behavior, often improving interpretability and controllability. Recent studies further explore case-specific rubric or criteria generation, where evaluation criteria are dynamically generated according to the current input (Gerner et al., 2026; Pathak et al., 2025). Although these methods demonstrate that evaluation criteria need not remain fully static, most existing approaches rely on free-form rubric generation and are not specifically designed for fine-grained QE scenarios. In translation quality evaluation, different cases often exhibit substantial variation in ambiguity, error density, and required exploration granularity. Free-form criteria generation may therefore lead to unstable coverage, inconsistent granularity, and uncontrolled evaluation behavior. Meanwhile, MQM-based QE methods still mainly rely on static rubric configurations, where all samples share the same subtype exploration space. In contrast, our work combines MQM' s structured subtype hierarchy with case-specific dynamic rubric allocation. Rather than freely generating arbitrary rubrics, the model dynamically selects and expands suitable MQM subtype spaces for each translation case.

Adaptive Routing and Dynamic Search Space Increasing the evaluation search space generally improves recall because the model can explore a broader range of potential error dimensions. However, exhaustive subtype exploration also tends to introduce large numbers of false positives and substantially higher inference cost. This suggests that different translation samples may require different exploration granularity: simpler cases may only need a compact subtype space, while more ambiguous or error-dense cases may benefit from broader exploration. Similar ideas have been widely studied in adaptive computation, mixture-of-experts routing, and test-time scaling, where models dynamically allocate computation budgets, reasoning paths, or expert activations according to input complexity (Graves, 2016; Shazeer et al., 2017; Fedus et al., 2022; Zhang et al., 2025). Recent test-time compute studies further emphasize that different instances naturally require different amounts of reasoning budget (Hu et al., 2025). However, existing routing methods mainly focus on token computation, expert activation, or reasoning-budget allocation, while relatively little attention has been paid to dynamically controlling the evaluation search space itself. Our work introduces routing into MQM-based QE to dynamically allocate rubric granularity and subtype exploration space for different translation cases, enabling more adaptive and case-specific error exploration.

3 Method

3.1 Task Formulation

Given a source sentence x and its machine translation y, the task is to detect erroneous spans in y. The output is a set of text spans:

$$\hat{\mathcal{S}} = \{\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_n\}, \tag{1}$$

where each $\hat{s}_i$ is a contiguous span copied from the translation.

Unlike sentence-level QE, span-level evaluation requires the model to localize concrete textual evidence. The exposed evaluation criteria therefore strongly affect the model's search behavior. Narrow rubrics may miss valid errors, whereas overly broad rubrics often introduce noisy detections.

We therefore formulate span-level QE as a case-specific MQM rubric activation problem. Instead of applying the same MQM configuration to all translations, the method dynamically selects relevant MQM-informed evaluation criteria for each source–translation pair. Figure 2 illustrates the overall cascaded evaluation framework. Given a source–translation pair, the framework first applies a correctness gate to filter cases that do not require further evaluation. For the remaining error-prone or uncertain cases, it then decides whether compact rubric evaluation is sufficient or whether broader MQM subtype exploration is needed. The final evaluation view is therefore selected case by case, ranging from a small rubric view to medium or full subtype rubric views.

3.2 Structured MQM Rubric Space

We construct a structured rubric space from the MQM taxonomy. Each MQM subtype is paired with a natural-language description and treated as an atomic rubric unit. A case-specific rubric is formed by selecting a subset of these units:

$$\mathcal{R}(x, y) \subseteq \mathcal{R}_{\text{MQM}}, \tag{2}$$

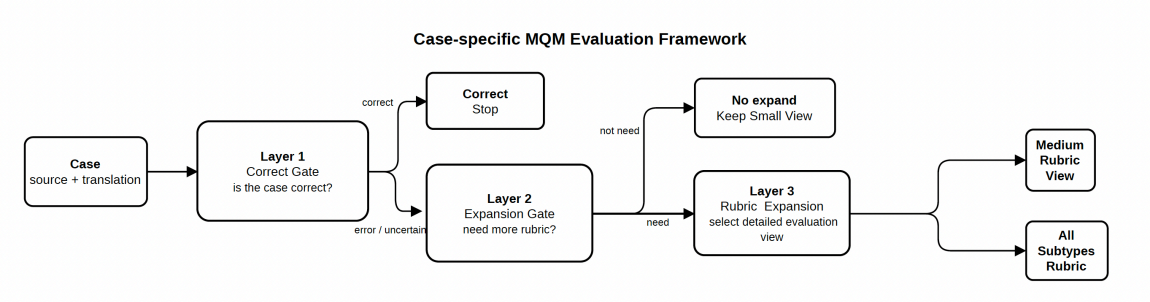

Figure 2: Overview of the case-specific MQM evaluation framework. The input case consists of a source sentence and its machine translation. The framework first uses a correctness gate to stop evaluation for cases predicted as correct. For error-prone or uncertain cases, an expansion gate determines whether the compact rubric view is sufficient. If further exploration is needed, the rubric expansion module selects a more detailed evaluation view, such as a medium-granularity rubric view or the full MQM subtype rubric view.

where $\mathcal{R}_{\mathrm{MQM}}$ denotes the complete MQM-derived rubric pool.

Unlike free-form rubric generation, the method does not invent new evaluation criteria. Instead, it dynamically selects from a predefined MQM-based rubric space, preserving MQM interpretability while adapting the exposed criteria to different translation cases.

3.3 Two-stage MQM Routing

Applying the full MQM subtype space to every translation is inefficient and often unnecessary. We therefore introduce a lightweight routing framework that first predicts relevant MQM major categories and then determines the required rubric granularity.

Major-category router. Given (x, y), we encode the source–translation pair using a Qwen3 causal language model and use the final hidden state as the sequence representation. A multi-label classification head predicts the probability of each MQM major category:

$$p_c = \sigma(w_c^\top h(x, y) + b_c), \tag{3}$$

where c denotes an MQM major category.

During inference, the router keeps the highest-scoring category and optionally activates additional categories according to score thresholds and relative confidence margins. The routing objective is recall-oriented activation rather than exact category matching.

Cascaded rubric routing. After category selection, the method dynamically determines how much subtype exploration is needed. A correctness gate first predicts whether the current translation can be accepted directly:

$$p_{\mathrm{ok}} = \sigma(w_{\mathrm{ok}}^\top h(x, y) + b_{\mathrm{ok}}). \tag{4}$$

If the translation is not accepted, the evaluator performs compact subtype evaluation using a small MQM subtype pool. An additional expansion gate then predicts whether broader rubric exploration is necessary:

$$p_{\mathrm{exp}} = \sigma(w_{\mathrm{exp}}^\top [h(x, y); e(x, y)] + b_{\mathrm{exp}}), \tag{5}$$

where $e(x, y)$ denotes compact-evaluation features.

Difficult cases are further expanded toward medium-granularity or full MQM subtype exploration. This cascaded design keeps the default inference path compact while allowing more complex translations to access broader rubric coverage.

3.4 Dynamic MQM-based Span Detection

Given the activated MQM categories and the selected rubric granularity, we retrieve the corresponding subtype descriptions and compose a case-specific evaluation rubric:

$$\hat{\mathcal{R}}(x, y) = \text{Top-}\hat{k}\big(\mathcal{R}_{\mathrm{MQM}}\big). \tag{6}$$

The composed rubric is inserted into the evaluation prompt together with the source sentence and translation. The downstream evaluator then performs span-level error detection under the activated MQM rubric context:

$$\hat{\mathcal{S}} = f_{\mathrm{eval}}(x, y, \hat{\mathcal{R}}(x, y)). \tag{7}$$

Finally, overlapping spans are merged and invalid predictions are filtered through lightweight

post-processing. The overall framework dynamically exposes different MQM evaluation spaces to different translation cases while preserving structured MQM-based evaluation behavior.

4 Experiments

4.1 Experimental Setup

Datasets. Experiments are conducted on the WMT23 span-level QE benchmark using Zh-En and En-De (Blain et al., 2023). The WMT23 development set is used for router training, and the test sets are used for evaluation. We mainly evaluate token-level span detection. MQM types and subtypes are used for rubric construction and prompting, but not as direct evaluation targets.

Models, training, and inference. We use Qwen3-4B and Qwen3-8B (Yang et al., 2025) in thinking mode with the vLLM backend (Kwon et al., 2023), and run all experiments on NVIDIA RTX A6000 GPUs. The routing modules are lightweight LoRA-based classifiers that predict the MQM search space from the source sentence and translation. The major-category router is trained on a frozen Qwen3-8B backbone with a linear multi-label head and LoRA adapters ($r = 4$, $\alpha = 8$, dropout 0.05) for 3 epochs, while the rubric-granularity router uses a slightly larger LoRA configuration ($r = 8$, $\alpha = 16$, dropout 0.05) and is trained for 6 epochs. Both routers use bf16 training. For span evaluation, we use deterministic decoding with `temperature`=0.0, `top_p`=1.0, and no sampling. Unless otherwise specified, vLLM inference uses maximum generation length 3072, maximum model length 8192, GPU memory utilization 0.90, and batch size 8 per GPU.

Evaluation metrics. We report word-level Precision, Recall, F1, and MCC. Predicted spans are converted into word-level labels and compared against gold error-span labels.

4.2 Baselines

We compare against baselines with different MQM rubric exposure strategies.

Direct prediction. The model directly predicts erroneous spans without fine-grained MQM subtype rubrics.

Static MQM rubric. All inputs are evaluated using the same predefined MQM rubric space. This baseline uses rubric-guided evaluation without instance-specific adaptation.

4.3 Experimental Variants

We compare three routing-based MQM rubric configurations. *Major-category routing* activates only the MQM major categories predicted for each instance. *Fixed-granularity routing* uses the selected major categories with predefined subtype spaces of different granularities. *Dynamic expansion* starts from a compact subtype space and selectively expands it for cases requiring finer-grained rubric coverage.

4.4 Main Results

Table 1 reports the word-level QE results on WMT23 Zh-En and En-De benchmarks.

Overall performance. The proposed framework consistently outperforms both direct Qwen3 baselines and previous QE systems across both language pairs. On Zh-En, Ours-8B achieves the best performance with 33.78 F1 and 32.40 MCC, substantially improving over DCSQE and CometKiwi. On En-De, Ours-8B also obtains the highest F1 and MCC, outperforming DCSQE by 2.33 F1 points and 2.46 MCC points. These results suggest that dynamically constructing case-specific MQM rubric spaces is more effective than direct prompting for span-level quality estimation.

Comparison with LLM baselines and recall improvements. Compared with vanilla Qwen3 models of the same scale, the proposed framework consistently yields large MCC improvements across both language pairs. On Zh-En, Ours-4B and Ours-8B improve over their corresponding Qwen3 baselines by 9.55 and 9.04 MCC points, respectively. Similar trends are also observed on En-De. In addition, the proposed method achieves substantially higher recall while maintaining competitive precision. On Zh-En, both Ours-4B and Ours-8B reach recall values close to 70%, compared with much lower recall from vanilla Qwen3 baselines. On En-De, Ours-8B further improves recall to 60.16%, leading to the overall best F1 and MCC results.

4.5 Ablation on Dynamic Rubric Routing

We further conduct ablation studies to analyze the contribution of different routing components. Table 2 reports results on the Zh-En WMT23 bench-

Method	WMT23 Zh-En				WMT23 En-De			
	P	**R**	**F1**	**MCC**	**P**	**R**	**F1**	**MCC**
CometKiwi	–	–	–	26.90	–	–	–	21.50
DCSQE	–	–	28.61	28.12	–	–	30.61	27.11
GEMBA	–	–	16.11	18.21	–	–	13.21	18.17
Qwen3-4B	17.04	45.79	24.84	19.70	16.31	16.19	16.25	10.23
Qwen3-8B	18.22	53.55	27.18	23.36	19.16	23.43	21.08	14.76
Qwen3-14B	21.94	44.78	29.45	24.52	23.99	20.83	22.30	17.16
Ours-4B	19.71	69.51	30.71	29.25	**31.75**	31.40	31.57	26.60
Ours-8B	**22.29**	**69.80**	**33.78**	**32.40**	22.68	**60.16**	**32.94**	**29.57**

Table 1: Word-level QE results on WMT23 Zh-En and En-De. All values are reported as percentages. Results of CometKiwi, DCSQE, and GEMBA are taken from prior work (Rei et al., 2022; Geng et al., 2025; Kocmi and Federmann, 2023).

Method	P	R	F1	MCC
Compact	14.80	65.63	24.16	21.48
Comprehensive	10.89	76.50	19.07	16.80
+ Correct Gate	13.70	70.45	22.95	20.74
+ Expansion Gate	18.93	71.79	29.96	28.88
Full Dynamic Routing	**22.29**	**69.80**	**33.78**	**32.40**

Table 2: Ablation results of dynamic rubric routing on Zh-En WMT23 using Qwen3-8B. Compact and Comprehensive correspond to fixed rubric configurations, while the remaining variants progressively introduce adaptive routing components.

Language Pair	**≤1 Cat.**	**≤2 Cats.**	**≤3 Cats.**
Zh-En	0.87	0.97	0.99
En-De	0.69	0.81	0.95

Table 3: Cumulative recall of effective errors covered by different numbers of activated MQM major categories.

mark using Qwen3-8B. Complete ablation results for all settings are provided in the appendix.

The results reveal a clear trade-off between compact and exhaustive rubric exploration. Fully comprehensive rubric spaces substantially improve recall, but also introduce many false positives, leading to lower precision and MCC. In contrast, compact rubric configurations maintain better precision but fail to sufficiently cover complex error cases.

The Correct Gate partially alleviates this issue by filtering likely error-free samples before rubric expansion, while the Expansion Gate further improves precision by selectively enlarging the evaluation search space only for difficult cases. These results support the core assumption of this work: different translation instances require different MQM evaluation search spaces, and dynamic rubric routing enables more effective allocation of MQM subtype exploration.

5 Analysis

5.1 Router Reliability Analysis

Before analyzing the final span-level predictions, we first examine whether the routing modules can provide a reliable evaluation search space. The proposed framework relies on two lightweight decisions before fine-grained span detection: the major-category router, which selects a small set of MQM major categories, and the no-error gate, which filters translations that are likely to contain no annotated error. These two components are designed to reduce unnecessary rubric exposure while preserving recall for error-bearing cases.

Major-category coverage. We first evaluate how many predicted MQM major categories are needed to cover the major categories corresponding to effective errors that can be identified during multi-category reasoning. Table 3 reports the cumulative recall when at most one, two, or three major categories are activated. On Zh-En, one major category already covers 87% of effective error cases, while allowing up to two categories increases the coverage to 97%. With at most three activated categories, the coverage further reaches 99%. On En-De, the distribution is more dispersed: one category covers 69%, two categories cover 81%, and three categories cover 95% of effective error cases.

These results show that **most effective errors can be covered by a small number of major categories**. Therefore, we use at most three predicted major categories in the final system. This setting **preserves high category-level recall while avoiding the noisy behavior caused by activat-**

Lang	Acc	G-OK	G-Bad	P-OK	P-Bad
Zh-En	0.522	1109	555	386	1278
En-De	0.649	1131	756	545	1342

Table 4: Statistics of the no-error gate. G/P denote gold/predicted labels.

ing the full MQM category space for every instance.

No-error gate. We further analyze the no-error gate based on the native thinking-mode prediction of the backbone model. Table 4 reports the gate accuracy and the distribution of gold and predicted labels. The gate obtains 52.22% accuracy on Zh-En and 64.88% accuracy on En-De. Although the overall accuracy is moderate, the gate is intentionally tuned toward retaining potentially erroneous cases rather than aggressively filtering them. In practice, about 95% of originally erroneous cases are kept for downstream rubric-based evaluation, so the recall of error-bearing instances is only minimally affected.

The remaining false-positive cases are not directly discarded at this stage. Instead, they are passed to the subsequent rubric-based evaluator, where case-specific MQM rubrics provide additional constraints for span identification. This design makes the gate recall-preserving: it mainly removes high-confidence no-error cases while leaving uncertain or potentially erroneous translations for fine-grained rubric inspection.

5.2 Dynamic Rubric Complexity Analysis

To better understand why dynamic rubric routing is necessary, we analyze how the required rubric granularity changes with translation complexity. Figure 3 and Figure 4 present the distribution of required rubric tiers on the Zh-En WMT23 benchmark using Qwen3-8B. Results for other settings are provided in the appendix.

We divide the evaluation search space into three rubric tiers: *Compact*, *Expanded*, and *Comprehensive*. Compact rubric sets correspond to a small number of highly relevant MQM subtypes, while Comprehensive settings expose a substantially larger fine-grained search space.

Figure 3 shows a **clear correlation between translation complexity and required rubric granularity**. As the number of error spans increases, the proportion of cases requiring Comprehensive rubric exploration rises steadily, while Compact configurations become less effective. A similar trend can be observed with source sentence length: longer and semantically denser inputs increasingly require larger evaluation search spaces.

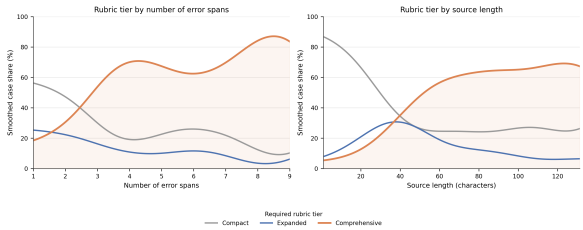

Figure 3: Smoothed distribution of required rubric tiers under different translation complexities. Left: trend with respect to the number of error spans. Right: trend with respect to source sentence length.

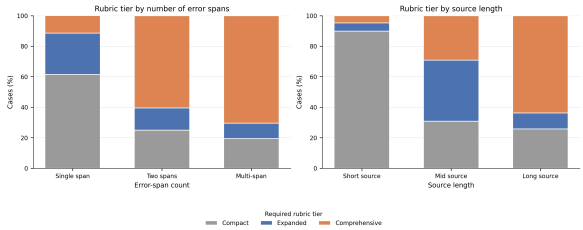

Figure 4: Bucketed statistics of required rubric tiers. Left: grouped by error-span count. Right: grouped by source sentence length.

Figure 4 further illustrates this phenomenon using coarse-grained buckets. For single-span cases and short inputs, Compact rubric settings dominate the distribution. In contrast, multi-span and long-source cases show a much larger proportion of Comprehensive routing decisions. The Expanded tier mainly appears in medium-complexity regions, acting as a transition between compact and exhaustive exploration.

These results support the central motivation of our framework: **different translation instances require substantially different evaluation search spaces**. Instead of applying a uniform rubric configuration to all samples, the router learns to adaptively allocate rubric granularity according to the estimated complexity of the current case. This dynamic allocation enables the evaluator to preserve recall on difficult samples while avoiding unnecessary search expansion on simpler cases.

5.3 Case Study

Table 5 presents a representative example comparing different rubric granularities. Error spans predicted by each setting are highlighted in red. We observe that different evaluation search spaces

Setting	Detected Spans	P	R	F1
Source	由于中西方文化背景等原因, 语言之间或多或少会存在差异, 如果有机会能看到作者原版相信会收获更多。	-	-	-
Reference	Due to the cultural context of the Midwest, there are more or less differences between the languages, and the author believes that if he gets a chance to see the original, he will reap more.	-	-	-
Baseline	Due to the cultural context of the Midwest, there are more or less differences between the languages, and the author believes that if he gets a chance to see the original, he will reap more.	1.000	0.111	0.200
Small Rubric	Due to the cultural context of the Midwest, there are more or less differences between the languages, and the author believes that if he gets a chance to see the original, he will reap more.	1.000	0.222	0.364
Medium Rubric	Due to the cultural context of the Midwest, there are more or less differences between the languages, and the author believes that if he gets a chance to see the original, he will reap more.	0.875	0.778	**0.824**
Full Rubric	Due to the cultural context of the Midwest, there are more or less differences between the languages, and the author believes that if he gets a chance to see the original, he will reap more.	0.310	1.000	0.474

Table 5: Case study under different rubric granularities.

lead to substantially different span discovery behaviors.

As shown in Table 5, different rubric granularities substantially influence the span exploration behavior of the evaluator. The baseline setting is relatively conservative and only detects the core mistranslation span, resulting in low recall. As the rubric search space expands, the model gradually identifies more semantically related error spans. The medium-rubric configuration achieves the best overall performance by covering the major error structures while maintaining relatively compact span boundaries. In contrast, the full-rubric setting further increases recall but also introduces substantially noisier spans and boundary errors.

6 Conclusion

This paper studies the role of MQM rubrics in LLM-based span-level translation quality evaluation and argues that rubrics function not only as explicit evaluation criteria, but also as evaluation search spaces that strongly influence the exploration behavior of large language models during error localization. Experimental analysis shows a clear relationship between rubric granularity and evaluation behavior: compact rubric spaces often improve precision but suppress useful exploration and reduce recall, while excessively large rubric spaces encourage broader error exploration at the cost of substantial false positives.

Based on this observation, we propose a case-specific dynamic rubric framework that dynamically allocates MQM evaluation granularity according to the complexity of individual translation instances. Instead of applying a fixed static rubric configuration to all samples, the proposed framework combines major-category routing, cascaded rubric expansion, and adaptive subtype selection to construct instance-level evaluation search spaces for span-level QE.

Experiments on WMT23 Zh-En and En-De benchmarks demonstrate that dynamic rubric allocation achieves substantially better overall evaluation performance than direct prediction and static MQM prompting. Further analysis reveals that different translation instances exhibit substantially different rubric granularity preferences, suggesting that evaluation variability in LLM-based QE is influenced not only by model capability itself, but also by how evaluation spaces are exposed to the evaluator. Dynamic rubric allocation provides a relatively lightweight mechanism for adapting to different data distributions and annotation preferences, enabling LLM evaluators to better express their underlying error detection capabilities under diverse evaluation conditions.

7 Limitations

Several limitations remain in the current work. First, the proposed framework still relies on supervised routing modules and heuristic granularity signals derived from MQM annotations. Although the method shows consistent improvements on WMT23 benchmarks, its robustness under different annotation standards, evaluation preferences, and unseen datasets remains insufficiently explored.

Second, the current study mainly focuses on

MQM-based span-level QE with Qwen-based models and a limited set of language directions. It remains unclear whether similar dynamic evaluation-space allocation strategies generalize to other LLM evaluators, multilingual settings, or broader structured evaluation tasks beyond translation quality estimation.

In addition, while the experiments suggest that evaluation-space exposure strongly affects exploration behavior during span detection, the interaction between rubric exposure, reasoning dynamics, and span generation is still largely empirical. The current framework dynamically adjusts rubric granularity, but does not explicitly model the underlying exploration process of LLM evaluators. Future work may further investigate exploration-aware decoding, adaptive evaluation-space control, and more general dynamic rubric mechanisms for structured LLM evaluation.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Catalin Anghel, Marian Viorel Craciun, Emilia Pecheanu, Adina Cocu, Adrian Istrate, and Andreea Alexandra Anghel. 2025. Rubric-guided evaluation framework for consistent scoring with large language models. In *2025 9th International Symposium on Electrical and Electronics Engineering (ISEEE)*, pages 1–6. IEEE.

Frederic Blain, Chrysoula Zerva, Ricardo Rei, Nuno M Guerreiro, Diptesh Kanojia, José GC de Souza, Beatriz Silva, Tânia Vaz, Yan Jingxuan, Fatemeh Azadi, and 1 others. 2023. Findings of the wmt 2023 shared task on quality estimation. In *Proceedings of the Eighth Conference on Machine Translation*, pages 629–653.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024. A survey on evaluation of large language models. *ACM transactions on intelligent systems and technology*, 15(3):1–45.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021. Experts, errors, and context: A large-scale study of human evaluation for machine translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Xiang Geng, Zhejian Lai, Jiajun Chen, Hao Yang, and Shujian Huang. 2025. Alleviating distribution shift in synthetic data for machine translation quality estimation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7546–7560.

Assaf Gerner, Netta Madvil, Nadav Barak, Alex Zaikman, Jonatan Liberman, Liron Hamra, Rotem Brazilay, Shay Tsadok, Yaron Friedman, Neal Harow, and 1 others. 2026. Deepchecks: Evaluating retrieval-augmented generation (RAG). *arXiv preprint arXiv:2605.14488*.

Alex Graves. 2016. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*.

Zhiwei He, Tian Liang, Wenxiang Jiao, Zhuosheng Zhang, Yujiu Yang, Rui Wang, Zhaopeng Tu, Shuming Shi, and Xing Wang. 2024. Exploring human-like translation strategy with large language models. *Transactions of the Association for Computational Linguistics*, 12:229–246.

Jinwu Hu, Zhitian Zhang, Guohao Chen, Xutao Wen, Chao Shuai, Wei Luo, Bin Xiao, Yuanqing Li, and Mingkui Tan. 2025. Test-time learning for large language models. *arXiv preprint arXiv:2505.20633*.

Ahrii Kim. 2025. RUBRIC-MQM: Span-level LLM-as-judge in machine translation for high-end models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 147–165.

Tom Kocmi and Christian Federmann. 2023. GEMBA-MQM: Detecting translation quality error spans with GPT-4. In *Proceedings of the Eighth Conference on Machine Translation*, pages 768–775.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 2511–2522.

Arle Lommel, Hans Uszkoreit, and Aljoscha Burchardt. 2014. Multidimensional quality metrics (mqm): A framework for declaring and describing translation quality metrics. *Tradumàtica*, (12):0455–463.

Aditya Pathak, Rachit Gandhi, Vaibhav Uttam, Arnav Ramamoorthy, Pratyush Ghosh, Aaryan Raj Jindal, Shreyash Verma, Aditya Mittal, Aashna Ased, Chirag Khatri, and 1 others. 2025. Rubric is all you need: Improving llm-based code evaluation with question-specific rubrics. In *Proceedings of the 2025 ACM Conference on International Computing Education Research V. 1*, pages 181–195.

Ricardo Rei, Marcos Treviso, Nuno M Guerreiro, Chrysoula Zerva, Ana C Farinha, Christine Maroti, José GC De Souza, Taisiya Glushkova, Duarte Alves, Luisa Coheur, and 1 others. 2022. COMETKiwi: IST-Unbabel 2022 submission for the quality estimation shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 634–645.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, and 1 others. 2025. A survey on test-time scaling in large language models: What, how, where, and how well? *arXiv preprint arXiv:2503.24235*.

A MQM Hierarchy and Prompt Templates

This appendix documents the MQM hierarchy used by the pipeline and the prompt templates used in the main inference variants. The templates below are shown before model-specific chat-template tokens are inserted. In the implementation, they are rendered with `tokenizer.apply_chat_template`; when thinking mode is enabled, the rendered prompt is additionally primed with `<think>Okay,`.

A.1 MQM Hierarchy

The full MQM taxonomy is organized as a tree. For this work, we use seven broad MQM major types as the top-level routing space. Table 6 shows representative flattened subtypes under each major type.

All lower-level nodes under each MQM major category are flattened into subtype candidates, where each subtype is associated with a short description from the original MQM hierarchy. For example, the Accuracy category contains subtypes such as Mistranslation and Number, while Linguistic conventions contains subtypes such as Grammar and Agreement. This flattened representation preserves the hierarchical semantic information while simplifying the inference interface.

In the proposed framework, the rubric is constructed by combining the model-predicted MQM subtype names with their corresponding descriptions. The model first predicts relevant MQM major categories for a given source–translation pair, and the selected categories are then expanded into description-enhanced subtype rubrics for span-level error detection. Some MQM categories, such as Locale conventions, contain multiple parent-linked entries including date format, currency format, and shortcut key conventions, which are also treated as flattened subtype candidates.

The hierarchy also provides concrete guidance for interpreting subtype behaviors. For instance, Number errors correspond to incorrect numeric values in translation, Overtranslation refers to targets that become unnecessarily more specific than the source, and Wrong term captures domain-specific terminology mismatches. Locale-specific content covers cases where the translation becomes inappropriate under the target locale, while Unidiomatic style describes grammatically correct but unnatural expressions. In addition, Truncation/text expansion refers to translations whose length no longer fits the intended layout or formatting constraints.

A.2 Baseline span prompts

Baseline span prompt

```
SYSTEM:
You are an MQM annotation assistant. Output
    JSON only, no explanation.
Required output schema: {"items":[{"
    error_text":"...","category":"..."]}
Rules:
1) error_text must be an exact text span
    copied from TARGET.
2) Do not output indexes; output only the
    erroneous text content.
3) category/severity must be strings.
4) If no error, output {"items":[]}.
```

Major type	Representative flattened subtypes
Accuracy	Mistranslation; Number; Date/time; Overtranslation; Omission
Audience appropriateness	Locale-specific content; Language-dependent logic; Offensive
Design and markup	Font; Kerning; Layout; Truncation/text expansion; Markup tag
Linguistic conventions	Grammar; Agreement; Word order; Punctuation; Spelling
Locale conventions	Date format; Currency format; Quote mark type; Shortcut key
Style	Language register; Awkward style; Unidiomatic style; Inconsistent style
Terminology	Wrong term; Inconsistent use of terminology; Organization terminology

Table 6: Seven MQM major types used as the top-level routing space, with representative flattened subtypes.

```
5) Never output extra keys or text.

USER:
Task: find erroneous text in TARGET and
    classify each with category + severity.
Return each error as exact copied text from
    TARGET.

SOURCE:
{source}

TARGET:
{target}
```

A.3 Merge Prompts

The merge pipeline first uses the major-category router to predict relevant MQM major types for each source–translation pair. For each predicted major type, all corresponding subtype candidates under that subtree are activated and used for span-level evaluation. The predicted spans and error types from different major-type subtrees are then merged into the final output.

The first prompt below is used by `format_hierarchy_subtree_predict_prompt`. In the standard merge variant, the minimal-span rule is disabled; in the minimal-span variant, the additional minimal-span line is inserted.

Hierarchy merge: major-subtree detection prompt

```
SYSTEM:
You are an MQM translation quality evaluation
     assistant. Output JSON only.

Return exactly one JSON object with the
    schema:
{"items":[{"error_text":"...","error_type
    ":"...","error_type_description":"..."]}

Task:
Evaluate the translation quality of TARGET
    against SOURCE.
Find translation errors in TARGET that belong
     to the provided MQM subtree.
If such errors exist, output one item for
    each detected error.

Rules:
- error_text must be exact copied text from
    TARGET.
- [optional for minimal-span runs] error_text
     must be the smallest TARGET span that
    still fully identifies the error; do not
     include surrounding correct text.
- error_type must be chosen exactly from
    CANDIDATE_CHECKS.error_type.
- error_type_description must exactly match
    the chosen candidate description.
- severity must be a string.
- If no matching error exists in this subtree
    , output {"items":[]}.
- Do not output extra keys or extra text.

USER:
MQM_TOP_LEVEL:
{top_level_name}

Instruction:
Compare SOURCE and TARGET and identify actual
     translation errors only.
Do not invent issues. But if a clear error
    exists in this MQM subtree, report it.

SOURCE:
{source}

TARGET:
{target}

CANDIDATE_CHECKS:
{candidate_checks_json}
```

After the subtree outputs are merged, the pipeline can run a second span inference pass guided by the merged predicted error types. This prompt is used by `format_eval_prompt_with_error_type_rubric`.

Hierarchy merge: merged-rubric span prompt

```
SYSTEM:
You are an MQM annotation assistant. Output
    JSON only, no explanation.
```

```
Required output schema: {"items":[{"
    error_text":"...","category":"..."]}
Rules:
1) error_text must be an exact text span
    copied from TARGET.
2) Do not output indexes; output only the
    erroneous text content.
3) category/severity must be strings.
4) If no error, output {"items":[]}.
5) Never output extra keys or text.

USER:
Task: identify translation errors in TARGET
    based on SOURCE.
Task: find erroneous text in TARGET and
    classify each with category + severity.
Return each error as exact copied text from
    TARGET.
Use rubric.checks as primary guidance for
    locating errors.
If no error is found, output {"items":[]}.

SOURCE:
{source}

TARGET:
{target}

RUBRIC_JSON:
{
  "scope": "sample",
  "checks": [
    {
      "mqm_error_type_display_name": "{
    predicted_mqm_type}",
      "error_type_description": "{description
    }",
      "rationale": "Use this predicted MQM
    type as guidance: {predicted_mqm_type}."
    }
  ]
}
```

A.4 Top-k Best Prompt

The top-k-best pipeline uses the seven-major subtype-ranking file to construct different granularities of candidate subtype sets for each major type. For each sample, the model is evaluated under progressively expanded MQM subtype spaces, ranging from compact subtype subsets to more comprehensive subtype collections. Each run uses the same minimal-span subtree prompt below, and the best output is selected by per-sample word-level F1.

Top-k-best minimal-span subtree prompt

```
SYSTEM:
You are an MQM translation quality evaluation
     assistant. Output JSON only.

Return exactly one JSON object with the
    schema:
{"items":[{"error_text":"...","error_type
    ":"...","error_type_description":"..."]}

Task:
Evaluate the translation quality of TARGET
    against SOURCE.
Find translation errors in TARGET that belong
     to the provided MQM subtree.
If such errors exist, output one item for
    each detected error.

Rules:
- error_text must be exact copied text from
    TARGET.
- error_text must be the smallest TARGET span
     that still fully identifies the error;
    do not include surrounding correct text.
- error_type must be chosen exactly from
    CANDIDATE_CHECKS.error_type.
- error_type_description must exactly match
    the chosen candidate description.
- severity must be a string.
- If no matching error exists in this subtree
    , output {"items":[]}.
- Do not output extra keys or extra text.

USER:
MQM_TOP_LEVEL:
{major_type}

Instruction:
Compare SOURCE and TARGET and identify actual
     translation errors only.
Do not invent issues. But if a clear error
    exists in this MQM subtree, report it.

SOURCE:
{source}

TARGET:
{target}

CANDIDATE_CHECKS:
{topk_candidate_checks_json}
```

A.5 Subtype-ranking Prompt

For each source–translation pair and MQM major category, the model predicts multiple granularities of subtype candidate sets. The goal of this stage is not to directly predict final translation errors, but to estimate which MQM subtype definitions would be most useful as auxiliary rubric entries for later span-level evaluation. Smaller candidate sets correspond to more compact rubric spaces, while larger candidate sets expose broader subtype exploration spaces.

Subtype-ranking prompt

```
SYSTEM:
You are an MQM rubric selection assistant.
Your final answer must be a JSON object with
    this schema:
```

```
{
  "compact":["subtype name"],
  "medium":["subtype name"],
  "large":["subtype name"],
  "full":["subtype name"]
}

Rules:
1) Select only subtype names from the
     provided SUBTYPE_RUBRIC_JSON.
2) Rank subtypes by usefulness as auxiliary
     rubric types for judging possible
     translation errors in this case.
3) The selected subtypes are not final
     activated error labels.
4) compact should contain the most essential
     subtype candidates.
5) medium should expand the compact subtype
     space with additional useful subtype
     candidates.
6) large should further expand the subtype
     space with broader subtype coverage.
7) full should contain the most comprehensive
      subtype exploration space available for
      this major category.
8) If the subtype inventory is small, return
     as many subtype names as available.
9) Do not include duplicate subtype names or
     labels outside the rubric.

USER:
Task: select MQM subtype rubric entries that
     would be most useful as auxiliary rubric
      types for judging possible translation
     errors in this SOURCE/TARGET pair.
Evaluate only the single major category shown
      in SUBTYPE_RUBRIC_JSON.
Do not decide the final error type. Rank
     subtype definitions by how helpful they
     are for analyzing this case.

SOURCE:
{source}

TARGET:
{target}

SUBTYPE_RUBRIC_JSON:
{
  "major_category": "{major_type}",
  "subtypes": [
    {
      "name": "{subtype_name}",
      "path": "{major_type}/{subtype_path}",
      "description": "{subtype_description}"
    }
  ]
}
```

B Detailed Method Description

B.1 Detailed MQM Category Routing

Let $\mathcal{C}$ denote the seven MQM major categories: Accuracy, Audience appropriateness, Design and markup, Linguistic conventions, Locale conventions, Style, and Terminology.

Given a source sentence x and translation y, we encode the pair using a Qwen3 causal language model with an instruction-style prompt. The router uses the final hidden state at the last non-padding token as the sequence representation:

$$h(x, y) = H^{(L)}_{\ell_i}, \qquad \ell_i = \sum_t m_{i,t} - 1, \tag{8}$$

where $H^{(L)}$ denotes the last-layer hidden-state matrix and $m_{i,t}$ is the attention mask.

A linear classification head predicts one score for each MQM category:

$$z_c = w_c^\top h(x, y) + b_c, \qquad p_c = \sigma(z_c), \tag{9}$$

where $c \in \mathcal{C}$.

The router is trained as a multi-label classifier using binary cross entropy:

$$\mathcal{L}_{\text{cat}} = -\sum_{c \in \mathcal{C}'} \left[\omega_c q_c \log p_c + (1 - q_c) \log(1 - p_c)\right], \tag{10}$$

where $\mathcal{C}'$ optionally includes a NoCategory label, $q_c \in \{0, 0.5, 1\}$ is a soft supervision signal derived from word-level span overlap, and ω_c is a positive-class reweighting factor estimated from the training split.

A category receives:

- $q_c = 1$ if the predicted span has strong overlap with a gold span;
- $q_c = 0.5$ for partial overlap;
- $q_c = 0$ otherwise.

To reduce imbalance between error-bearing and no-error cases, all-zero MQM examples are downsampled during training.

During inference, the router is tuned toward recall-oriented activation. Let

$$c_1 = \arg\max_{c \in \mathcal{C}} p_c \tag{11}$$

be the highest-scoring category. Additional categories are activated if

$$p_c \geq \tau_{\text{cat}} \quad \text{or} \quad p_c \geq p_{c_1} - \delta, \tag{12}$$

up to a maximum number of activated categories.

If the optional NoCategory score is high while all MQM category scores remain below a minimum confidence threshold, the router returns an empty category set. For extremely low-confidence cases, Accuracy is used as a fallback category because it covers common adequacy-related failures.

B.2 Detailed Cascaded Rubric Routing

After MQM major-category selection, the framework dynamically determines how much subtype exploration is needed.

Correctness gate. A correctness gate first predicts whether the translation can be accepted without additional subtype evaluation:

$$a_{\text{ok}} = w_{\text{ok}}^{\top} h(x, y) + b_{\text{ok}}, \tag{13}$$

$$p_{\text{ok}} = \sigma(a_{\text{ok}}). \tag{14}$$

If $p_{\text{ok}} \geq \tau_{\text{ok}}$, the system directly returns a no-error decision.

Otherwise, the evaluator proceeds with compact subtype evaluation using a small set of highly relevant MQM subtypes.

Expansion gate. Some translations require broader rubric coverage beyond compact evaluation. We therefore introduce an expansion gate after compact evaluation.

Let $e_3(x, y)$ denote compact-evaluation features. The expansion gate predicts whether broader subtype exploration is necessary:

$$a_{\text{exp}} = w_{\text{exp}}^{\top} [h(x, y); e_3(x, y)] + b_{\text{exp}}, \tag{15}$$

$$p_{\text{exp}} = \sigma(a_{\text{exp}}). \tag{16}$$

If $p_{\text{exp}} < \tau_{\text{exp}}$, the compact evaluation result is kept. Otherwise, a budget router determines whether to expand toward a medium-granularity subtype pool or the full subtype pool:

$$\mathcal{B} = \{\text{medium}, \text{full}\}, \tag{17}$$

$$\hat{b} = \arg\max_{b \in \mathcal{B}} \pi_b(x, y, e_3). \tag{18}$$

The medium setting exposes a broader but still selective subtype space, whereas the full setting activates all available subtypes under the selected MQM categories.

B.3 Routing Supervision

To supervise routing behavior, we evaluate each candidate granularity and define a utility function:

$$U(g) = F_1(g) - \lambda \, \text{cost}(g), \tag{19}$$

where $F_1(g)$ denotes word-level F1 under granularity setting g.

The oracle granularity is defined as:

$$g^{\star} = \arg\max_{g \in \{\text{small}, \text{medium}, \text{full}\}} U(g). \tag{20}$$

The correctness gate is trained using binary supervision:

$$\mathcal{L}_{\text{ok}} = \text{BCEWithLogits}(a_{\text{ok}}, y_{\text{ok}}). \tag{21}$$

The expansion gate predicts whether oracle granularity exceeds compact evaluation:

$$y_{\text{exp}} = \mathbb{I}[g^{\star} \neq \text{small}], \tag{22}$$

$$\mathcal{L}_{\text{exp}} = \text{BCEWithLogits}(a_{\text{exp}}, y_{\text{exp}}). \tag{23}$$

The budget router is trained only on examples requiring expansion:

$$\mathcal{L}_{\text{bud}} = \mathbb{I}[g^{\star} \neq \text{small}] \, \text{CE}\left(\pi, \text{index}_{\mathcal{B}}(g^{\star})\right). \tag{24}$$

The total routing objective is:

$$\mathcal{L} = \mathcal{L}_{\text{cat}} + \mathcal{L}_{\text{ok}} + \mathcal{L}_{\text{exp}} + \mathcal{L}_{\text{bud}}. \tag{25}$$

B.4 Detailed Dynamic Span Detection

Given activated categories $\hat{\mathcal{C}}(x, y)$ and rubric size $\hat{k}$, the framework retrieves the corresponding subtype descriptions from the MQM rubric space:

$$\hat{\mathcal{R}}(x, y) = \text{Top-}\hat{k}\left(\{r \in \mathcal{R}_{\text{MQM}} : \text{Major}(r) \in \hat{\mathcal{C}}(x, y)\}\right). \tag{26}$$

The composed rubric is inserted into the evaluation prompt together with the source sentence and translation:

$$\hat{\mathcal{S}} = f_{\text{eval}}(x, y, \hat{\mathcal{R}}(x, y)). \tag{27}$$

The evaluator is instructed to output only spans copied from the translation, ensuring alignment with the span-level QE objective.

When multiple evaluation outputs are produced, we apply lightweight post-processing:

- highly overlapping spans are merged;
- duplicated predictions are removed;
- invalid spans not appearing in the translation are filtered;
- optional calibration combines think and no-think outputs.

This design allows different translation cases to access different MQM subtype spaces while preserving structured MQM-based evaluation behavior.

C Ablation Settings

To better understand the contribution of each component in the proposed framework, we conduct a series of ablation experiments with progressively increasing levels of dynamic routing capability.

We replace the original fixed top-k terminology with three semantic search-space levels:

- **Compact**: a small fine-grained rubric space;
- **Expanded**: a medium-sized rubric space;
- **Comprehensive**: the full MQM subtype space.

The evaluated settings are summarized in Table 7.

The **Correct Gate** predicts whether a translation is likely to contain critical errors. Samples predicted as correct terminate early without fine-grained evaluation. Otherwise, the system enters Compact evaluation.

The **Expansion Gate** determines whether the current Compact search space is sufficient. If not, the evaluation space is expanded to a larger rubric set.

Finally, the **Budget Router** controls the expansion granularity for difficult cases, dynamically selecting between Expanded and Comprehensive search spaces according to the estimated sample complexity.

C.1 Ablation Results

Tables 8 and 9 present the complete ablation results on WMT23 Zh→En and En→De benchmarks. We report word-level Precision (P), Recall (R), F1, and MCC.

Overall, the results show that simply enlarging the evaluation search space from Compact to Comprehensive mainly improves recall, but often introduces substantial false positives and reduces precision. The Correct Gate improves precision by filtering likely error-free samples before detailed evaluation. The Expansion Gate further improves the balance between precision and recall by selectively enlarging the search space only for difficult cases.

Finally, the Full Dynamic configuration consistently achieves the strongest overall performance across both language directions and model scales, demonstrating the effectiveness of instance-level adaptive rubric routing.

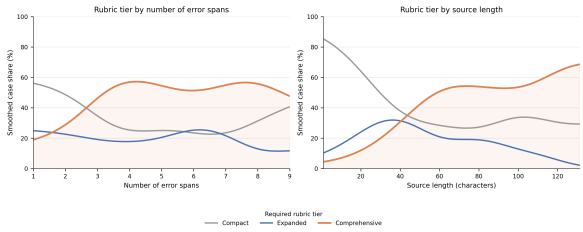

Figure 5: Smoothed distribution of required rubric tiers under different translation complexities on ZH-EN 2023 with Ours-4B. Left: trend with respect to the number of error spans. Right: trend with respect to source sentence length.

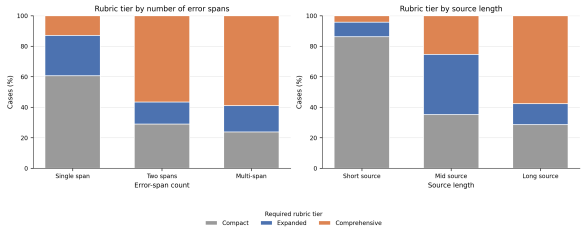

Figure 6: Bucketed statistics of required rubric tiers on ZH-EN 2023 with Ours-4B. Left: grouped by error-span count. Right: grouped by source sentence length.

D Additional Dynamic Rubric Complexity Analysis

We provide additional analysis of the relationship between translation complexity and required rubric granularity. Following the main analysis, we divide the evaluation search space into three rubric tiers: *Compact*, *Expanded*, and *Comprehensive*. The additional results cover ZH-EN with Ours-4B, EN-DE with Ours-4B, and EN-DE with Ours-8B.

Overall, the additional results show trends consistent with the main analysis. Cases with fewer error spans or shorter source sentences are more often handled by Compact rubric settings, while multi-span or longer inputs require larger rubric spaces more frequently. The Expanded tier usually appears in medium-complexity regions, serving as an intermediate level between compact and comprehensive exploration.

These results further support the need for case-specific rubric allocation. A fixed rubric configuration cannot equally match the evaluation needs of simple and complex translation instances, whereas dynamic routing allows the evaluator to adjust the exposed MQM search space according to sample complexity.

Setting	Correct	Expand	Budget	Output Space
Fixed Compact	×	×	×	Compact
Fixed Expanded	×	×	×	Expanded
Fixed Full	×	×	×	Comprehensive
+ Correct Gate	✓	×	×	stop / Compact
+ Expansion Gate	✓	✓	×	stop / Compact / Expanded
Full Dynamic	✓	✓	✓	adaptive full routing

Table 7: Ablation settings for dynamic rubric routing.

Model	Setting	P	R	F1	MCC
Ours-4B	Fixed Compact	13.74	68.53	22.89	20.39
	Fixed Expanded	14.14	70.59	23.56	20.59
	Fixed Comprehensive	11.99	73.86	20.63	18.26
	+ Correct Gate	15.86	63.85	25.41	22.64
	+ Expansion Gate	17.56	75.71	28.51	28.04
	Full Dynamic	**19.71**	69.51	**30.71**	**29.25**
Ours-8B	Fixed Compact	14.80	65.63	24.16	21.48
	Fixed Expanded	15.40	65.99	24.98	22.48
	Fixed Comprehensive	10.89	76.50	19.07	16.80
	+ Correct Gate	13.70	70.45	22.95	20.74
	+ Expansion Gate	18.93	71.79	29.96	28.88
	Full Dynamic	**22.29**	69.80	**33.78**	**32.40**

Table 8: Ablation results on WMT23 Zh→En.

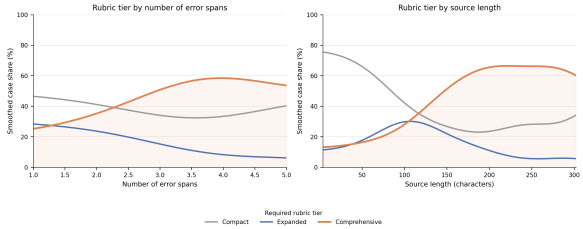

Figure 7: Smoothed distribution of required rubric tiers under different translation complexities on EN-DE 2023 with Ours-4B. Left: trend with respect to the number of error spans. Right: trend with respect to source sentence length.

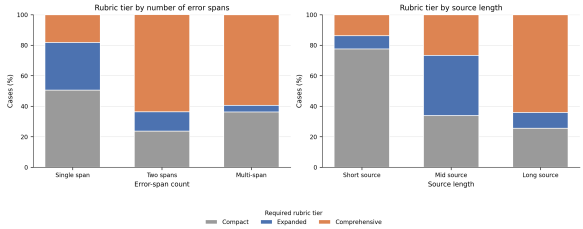

Figure 8: Bucketed statistics of required rubric tiers on EN-DE 2023 with Ours-4B. Left: grouped by error-span count. Right: grouped by source sentence length.

E Rubric Search Space Effects

Our experiments suggest that MQM rubrics function not only as explicit evaluation criteria, but also as a form of evaluation search space that influences the exploration behavior of the evaluator.

Given a source sentence x, translation y, and rubric space $\mathcal{R}$, the span prediction process can be formulated as:

$$\hat{\mathcal{S}} = f(x, y, \mathcal{R}), \tag{28}$$

where $\mathcal{R}$ denotes the currently activated MQM subtype space.

During free-form span prediction, the model often outputs both error spans and their associated MQM subtypes simultaneously. A natural assumption is therefore that the predicted subtypes already capture the model's internal error hypotheses, allowing the evaluation space to be compressed into:

$$\mathcal{R}_{\text{compact}} \subset \mathcal{R}_{\text{full}}. \tag{29}$$

Under this assumption, one may expect that retaining only the predicted subtypes should preserve most valid error spans while removing irrelevant rubric branches. In practice, however, we observe the opposite behavior. Empirical results show that aggressively restricting the rubric space substantially reduces recall. On WMT23 Zh-En using Qwen3-8B, the word-level recall drops from approximately 0.70 to 0.46 when only predicted subtypes are retained.

This phenomenon indicates that MQM rubrics influence not only the final labeling stage, but

Model	Setting	P	R	F1	MCC
Ours-4B	Fixed Compact	14.71	38.30	21.26	14.28
	Fixed Expanded	13.64	43.59	20.77	14.52
	Fixed Comprehensive	14.09	42.13	21.11	14.73
	+ Correct Gate	19.66	36.19	22.46	16.10
	+ Expansion Gate	26.72	30.26	28.38	22.83
	Full Dynamic	**31.75**	31.40	**31.57**	**26.60**
Ours-8B	Fixed Compact	13.00	45.96	20.27	13.81
	Fixed Expanded	12.67	46.11	19.87	13.33
	Fixed Comprehensive	14.18	48.09	21.90	16.01
	+ Correct Gate	18.18	45.09	22.90	18.01
	+ Expansion Gate	25.16	30.10	27.41	21.68
	Full Dynamic	**22.68**	**60.16**	**32.94**	**29.57**

Table 9: Ablation results on WMT23 En→De.

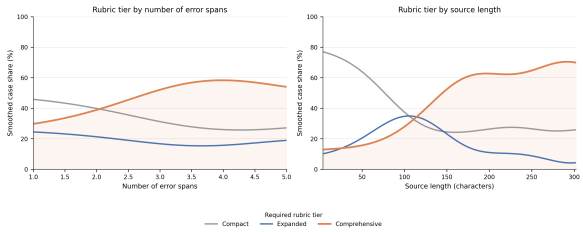

Figure 9: Smoothed distribution of required rubric tiers under different translation complexities on EN-DE 2023 with Ours-8B. Left: trend with respect to the number of error spans. Right: trend with respect to source sentence length.

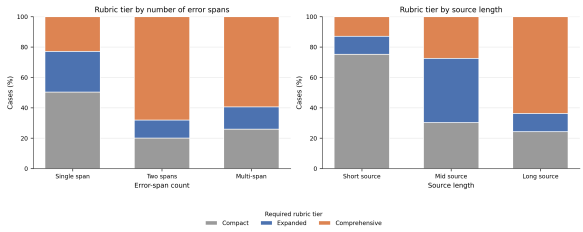

Figure 10: Bucketed statistics of required rubric tiers on EN-DE 2023 with Ours-8B. Left: grouped by error-span count. Right: grouped by source sentence length.

also the intermediate reasoning and exploration dynamics of the evaluator:

$$\mathcal{R} \rightarrow \text{reasoning paths} \rightarrow \hat{\mathcal{S}}. \tag{30}$$

A larger rubric space appears to encourage the model to inspect translations from multiple semantic perspectives simultaneously. Different MQM subtypes may trigger partially overlapping but distinct reasoning trajectories, including adequacy checking, fluency verification, terminology consistency inspection, stylistic comparison, or structural alignment analysis. Even when a subtype is not ultimately selected in the final output, its presence in the rubric space may still activate useful intermediate reasoning behaviors that help expose additional candidate spans.

We further observe that many valid spans discovered under larger rubric configurations cannot be reproduced after the rubric space is compressed, even when the final predicted subtype remains unchanged. This suggests that the missing spans are not caused purely by incorrect subtype prediction. Instead, restricting the rubric space alters the evaluator's decoding behavior itself and suppresses alternative exploration paths during generation.

The effect becomes particularly visible for ambiguous or weakly localized translation errors. Some spans may simultaneously admit multiple interpretations under different MQM perspectives. For example, a mistranslation may also exhibit stylistic awkwardness or terminology inconsistency. When only a narrow subset of subtypes is exposed, the evaluator tends to converge prematurely toward a limited interpretation space, reducing the probability of exploring alternative candidate spans.

This observation also helps explain why fully static large rubrics often achieve strong recall despite introducing substantial noise. Broad rubric spaces implicitly enlarge the evaluator's exploration frontier, allowing the model to retain more potentially useful hypotheses during generation. The improvement therefore does not solely originate from accurate subtype matching, but also from exposing diverse error perspectives to the evaluator.

At the same time, excessively large rubric spaces may introduce irrelevant reasoning branches and unstable predictions. The role of

dynamic routing is therefore not simply to minimize the rubric space, but to allocate a sufficiently expressive evaluation space for each translation instance. Simple cases may only require compact rubric subsets, while difficult or ambiguous cases benefit from broader MQM exploration spaces.

Overall, these results suggest that MQM rubrics should be viewed as dynamic reasoning constraints rather than static label inventories. Effective span-level QE depends not only on which MQM categories are selected, but also on how the selected rubric space shapes the evaluator's exploration behavior during inference.