

IFMTBench: A Comprehensive Benchmark for Multilingual Translation Instruction Following

Mingrui Sun*, Mao Zheng*, Zheng Li*, Mingyang Song

Large Language Model Department, Tencent

{akihitosun, moonzheng, jasonzli, nickmysong}@tencent.com

Abstract

Modern translation workflows demand more than semantic equivalence. Users routinely require models to preserve JSON or HTML schemas, honor curated glossaries, disambiguate with provided context, and match prescribed registers, often several at once. Conventional metrics such as BLEU and xCOMET capture semantic fidelity but provide little signal on constraint adherence, while general instruction following benchmarks ignore the cross-lingual nature of translation. We introduce IFMTBench, a benchmark for multilingual translation instruction following covering seven high-resource languages, with 4,506 single-constraint and 2,838 multi-constraint items spanning seven constraint dimensions and five compositional patterns with instructions issued in all seven languages. Constraints are split into a gating subset verified by deterministic checkers and a continuous subset scored by a rubric-based LLM judge, combined under a multiplicative rule that resists reward hacking. Evaluating 15 models reveals systematic gaps that prior protocols miss: Instruction following scales with size more sharply than translation quality, glossary and structured-format constraints dominate the difficulty gradient, and general instruction following rankings correlate only weakly with translation behavior. Our benchmark is available at <https://github.com/Tencent-Hunyuan/Hy-MT2/tree/main/IFMTBench>.

1 Introduction

A production translation request rarely consists of a bare source sentence. A typical request might ask a model to translate a marketing payload from English to Japanese while preserving an HTML schema, honoring a domain glossary, keeping inline code spans untranslated, and matching a prescribed register, all in a single response. Large

language models have absorbed this expanded role and turned translation from a sentence-level mapping task into a cross-lingual instruction following workflow (Llama Team, AI @ Meta, 2024; Yang et al., 2025; Gemini Team, Google, 2024). As such workflows are increasingly used to construct reward signals for post-training methods such as RLHF and GRPO (Ouyang et al., 2022; Shao et al., 2024; Rafailov et al., 2023), high-quality multi-dimensional evaluation of translation-specific instruction following has become acute.

Existing evaluation protocols leave a sizable blind spot. Conventional metrics such as BLEU (Papineni et al., 2002), chrF (Popović, 2015), and learned metrics including COMET (Rei et al., 2020), xCOMET (Guerreiro et al., 2024), and GEMBA (Kocmi and Federmann, 2023) are designed around semantic equivalence with a reference. Under these metrics, a fluent but constraint-violating output—one that silently drops a required glossary term or breaks a JSON schema—can score indistinguishably from a fully faithful translation. General instruction following benchmarks such as IFEval (Zhou et al., 2023), FollowBench (Jiang et al., 2024), and InFoBench (Qin et al., 2024) probe constraint adherence in depth but assume a monolingual setting. Multi-IF (He et al., 2024) extends IFEval-style verifiable instructions to multiple languages, yet inherits IFEval’s task formulation and is not designed around translation-specific constraints.

We present IFMTBench (Instruction Following for Machine Translation), a benchmark that addresses this gap along four axes. First, we formalize the translation constraint space into two mutually exclusive subsets, hard gating constraints with deterministic 0/1 verifiers and soft continuous constraints scored on a $[0, 1]$ rubric. Second, we curate 4,506 single-constraint and 2,838 multi-constraint items covering seven core dimensions and five compositional patterns, across seven high-resource tar-

*Equal contribution.

get languages (Chinese, English, German, French, Japanese, Korean, and Spanish). Third, we design a hybrid evaluation pipeline that pairs rule-based checkers for gating constraints with a rubric-conditioned LLM judge for continuous ones, aggregated under a multiplicative scoring rule that prevents soft scores from masking hard-constraint failures. Fourth, we report a two-dimensional view that decouples instruction following from translation quality, measured by xCOMET-XXL.

Evaluating 15 open- and closed-weight models on IFMTBench surfaces findings not visible under prior protocols. Instruction following ability scales much more sharply with model size than translation quality does, and the gap is amplified by an order of magnitude under multi-constraint composition. Glossary adherence and structured-format preservation dominate the difficulty gradient, while layout and code-keep constraints remain near-saturated. Translation quality and instruction following are only loosely coupled, and rankings on general instruction following benchmarks correlate strongly with IFMTBench only when comparisons are dominated by model size, with Spearman dropping to 0.65 (IFEval) and 0.55 (IFBench) among top-tier models.

Our contributions are: (i) a formal taxonomy of translation constraints with a multiplicative gating-plus-rubric scoring rule aligned with industrial quality requirements, (ii) a multilingual dataset of 7,344 expert-verified items spanning seven constraint dimensions, five composition patterns, and seven languages, with instructions paraphrased in all seven languages, (iii) a hybrid evaluation framework pairing deterministic checkers with a rubric-conditioned LLM judge and xCOMET-XXL for a two-dimensional view, and (iv) an empirical study of 15 models exposing diagnostic signals for translation-specific post-training alignment.

2 Related Work

Translation Quality Evaluation. Machine translation evaluation has evolved from n -gram overlap to neural semantic modeling. BLEU (Papineni et al., 2002) and chrF (Popović, 2015) measure lexical and character-level agreement with references. Learned neural metrics such as COMET (Rei et al., 2020) and xCOMET (Guerreiro et al., 2024) leverage cross-lingual representations to predict human judgments, and GEMBA (Kocmi and Federmann, 2023) uses GPT-4-class models as reference-free

quality estimators. All of these target *semantic equivalence*, so a fluent translation that violates auxiliary instructions, for example by dropping a glossary term or breaking a JSON schema, can still receive a high score, rendering such violations invisible to the evaluation protocol.

General Instruction Following Benchmarks.

IFEval (Zhou et al., 2023) constructs verifiable instructions checkable by deterministic scripts, FollowBench (Jiang et al., 2024) organizes constraints into five difficulty levels, InFoBench (Qin et al., 2024) decomposes prompts into atomic information requirements, and IFBench (Pyatkin et al., 2025) targets generalization to out-of-domain verifiable constraints. These benchmarks assume instruction and response share the same language, almost always English, and therefore do not exercise the cross-lingual semantic preservation that defines translation.

Translation Meets Instruction Following.

Work at this intersection is still nascent. WMT shared tasks (Kocmi et al., 2023) remain the central translation venue but place limited emphasis on user-imposed constraints. Constraint-aware translation has been studied for individual dimensions, such as terminology control via prefix conditioning (Dinu et al., 2019), but existing efforts typically address one constraint type in isolation and lack a unified multi-dimensional framework. Multi-IF (He et al., 2024) is the closest prior work, extending IFEval’s paradigm to multiple languages, yet its constraint set is inherited from IFEval rather than tailored to translation, and dimensions central to industrial workflows (glossary adherence, structured-format preservation, layout retention, and context-conditioned disambiguation) are largely absent or only indirectly covered. IFMTBench fills this gap with a translation-native constraint taxonomy, multiplicative gating-plus-rubric scoring, multilingual coverage, and a two-dimensional view that pairs instruction following with translation quality.

3 Constraint Taxonomy

A central premise of IFMTBench is that translation constraints should be evaluated under a structure that mirrors how they fail in practice. We partition the overall constraint space $\mathcal{C}$ into two mutually exclusive subsets, gating constraints $\mathcal{C}_{\text{gate}}$ and continuous constraints $\mathcal{C}_{\text{cont}}$. Gating constraints are de-

terministically verifiable and binary in nature, for example whether a required glossary term appears in the output or whether a JSON schema remains parseable. Continuous constraints, by contrast, concern soft properties such as register alignment or fidelity to auxiliary context on a $[0, 1]$ scale.

3.1 Core Constraint Dimensions

IFMTBench covers seven core constraint dimensions, organized into three macroscopic groups.

Lexical and Semantic. **Glossary** constraint $\mathcal{C}_{\text{glossary}} \in \mathcal{C}_{\text{gate}}$ requires the translation to contain specified target-language terms, possibly with sense disambiguation among multiple candidate translations. **Context** constraint $\mathcal{C}_{\text{ctx}} \in \mathcal{C}_{\text{cont}}$ requires the translation to be coherent with externally provided background that disambiguates polysemy or supplies cultural and domain information.

Structure and Format. **Structured-data** constraint $\mathcal{C}_{\text{struct}} \in \mathcal{C}_{\text{gate}}$ requires the translation to preserve JSON, HTML, CSV, or Markdown schemas, with topology and key names intact. **Layout** constraint $\mathcal{C}_{\text{layout}} \in \mathcal{C}_{\text{gate}}$ requires that specific separators and placeholders be preserved verbatim, **code-keep** constraint $\mathcal{C}_{\text{ckeep}} \in \mathcal{C}_{\text{gate}}$ requires that inline code spans remain untranslated, and **code-tag** constraint $\mathcal{C}_{\text{ctag}} \in \mathcal{C}_{\text{gate}}$ requires that code-tag markers wrapping protected spans must be retained unchanged.

Style and Register. **Style** constraint $\mathcal{C}_{\text{style}} \in \mathcal{C}_{\text{cont}}$ requires the translation to align with a prescribed formality level or tone.

3.2 Composition of Multiple Constraints

Real user prompts typically impose several constraints simultaneously. IFMTBench includes 2,838 multi-constraint items drawn from five recurring patterns observed in industrial workflows:

- context + glossary + style.
- glossary + struct + style.
- glossary + style.
- glossary + struct.
- context + glossary.

The selection follows two principles. The first is *representativeness*, since these patterns dominate game localization, software internationalization, and marketing content workflows. The second is

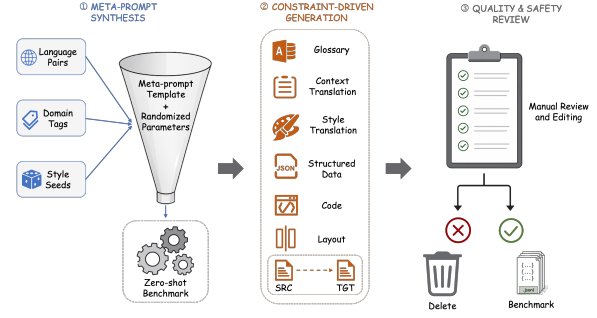


Figure 1: Three-stage data construction pipeline for IFMTBench. (1) Meta-prompt synthesis samples language pairs, domain tags, and style seeds with randomized parameters. (2) Constraint-driven generation instantiates prompts under seven dimensions of translation constraints. (3) Expert review verifies constraint satisfaction, rewrites unnatural phrasings, and performs a safety pass.

compatibility, since certain combinations are intrinsically conflicting (for example asking a model to both parse a JSON schema and obey a custom delimiter scheme) and are rare in practice, so we exclude them.

For each prompt p with constraint set $\mathcal{C}_p$, let $\mathcal{G}_p = \mathcal{C}_p \cap \mathcal{C}_{\text{gate}}$ and $\mathcal{S}_p = \mathcal{C}_p \cap \mathcal{C}_{\text{cont}}$. We aggregate per-constraint scores into the prompt-level instruction following score:

$$IF_Score(p) = \underbrace{\left(\prod_{c_g \in \mathcal{G}_p} s(c_g) \right)}_{\text{gating product}} \times \underbrace{\left(\frac{1}{|\mathcal{S}_p|} \sum_{c_s \in \mathcal{S}_p} s(c_s) \right)}_{\text{continuous mean}}, \quad (1)$$

where $s(c_g) \in \{0, 1\}$ for gating constraints and $s(c_s) \in [0, 1]$ for continuous ones, and by convention an empty product or mean evaluates to one. The multiplicative form grants any single gating failure veto power over the entire prompt, since a translation that breaks the structural contract is unusable regardless of how well it matches the requested style. In contrast, additive averaging, common in earlier instruction following benchmarks, can mask hard-constraint violations behind strong soft-constraint scores and is therefore exploitable when used as a reward signal for post-training.

4 Dataset Construction

IFMTBench is built through a three-stage pipeline that combines constraint-driven synthesis with expert human verification. Figure 1 summarizes the whole data construction flow, and the parallel evaluation pipeline is detailed in Section 5 and visualized in Figure 2.

4.1 Constraint-Driven Synthesis

We adopt a constraint-first meta-prompting approach in which the synthesizer is conditioned on

the target constraint specification before any source text is produced. Unlike pipelines that attach constraints post-hoc onto an existing corpus, our generator first internalizes the constraint, for example a specific polysemous term with its disambiguating context or a partial glossary table, and only then drafts a source sentence whose semantics naturally invite that constraint. This ordering helps avoid mismatches such as glossary terms that are pragmatically incompatible with their surrounding context.

To prevent mode collapse, the meta-prompt injects multiple stochastic factors at synthesis time, including a domain tag (e.g., game UI, marketing copy, technical documentation), a style seed (formal, casual, archaic, etc.), a target length range, and the source-target language pair. The model is forced to switch context across calls, yielding broad coverage along the domain and stylistic axes.

4.2 Expert Human Verification

To address known concerns about synthetic data quality, every item undergoes manual review by professional linguists. Reviewers (i) rewrite unnatural “model-style” phrasings so that both source and reference read as native text, (ii) verify that the reference translation in fact satisfies every declared constraint, (iii) for context-conditioned items, confirm that the reference is consistent with the supplied background, and (iv) perform a safety pass that removes sensitive content and de-identifies synthetic names, addresses, and phone numbers.

How much of the raw model output survives?

We logged the outcome of every review decision, since the answer determines whether “LLM generation with expert verification” is an accurate description of the pipeline. Table 1 reports the distribution. Slightly under half of all items (45.09%) received at least a light edit, and multi-constraint items—intrinsically harder to construct—show a systematically higher edit rate than single-constraint ones, indicating that review performs substantive work rather than rubber-stamping. Edit rate is nonetheless only an indirect proxy for whether the surviving text carries a detectable model signature: an unedited draft may read as natural, while a lightly edited item may retain formulaic phrasing. We therefore also asked reviewers to flag detectably template-like phrasing (stock openers, boilerplate constructions, unnaturally uniform rhythm) independently of edit status on a 600-item spot sample;

Review outcome	Share
Unchanged (draft accepted as-is)	54.91%
Light edit (constraint content kept)	40.99%
Substantial rewrite	4.10%

Table 1: Expert review outcomes over the full release. Light edits touch wording and naturalness only; substantial rewrites materially change the source and/or reference.

8.3% of items were flagged.

After this pipeline, IFMTBench contains 4,506 single-constraint items and 2,838 multi-constraint items across seven target languages.

4.3 Multilingual Design

Target languages. IFMTBench covers Chinese, English, German, French, Japanese, Korean, and Spanish. The selection balances three criteria. The first is *language-family diversity* (Sino-Tibetan, Germanic, Romance, Japonic, and Koreanic families). The second is *industrial relevance*, since these languages correspond to the top global software localization markets. The third is *writing-system diversity* (Latin script, Chinese characters, mixed kana and kanji, and Hangul), which directly stresses constraints such as code-tag retention and structured-format preservation under different character spaces.

Translation directions. Each item declares a fixed source-target language pair, and the overall dataset spans many-to-many directions among the seven languages, rather than collapsing onto an English- or Chinese-centric hub.

Instruction-language diversity. A distinctive design choice is to vary the language of the instruction itself while holding the source and target fixed. Each constrained translation task is paraphrased into instructions written in all seven languages, yielding parallel instruction sets that isolate instruction-parsing ability from translation competence. This decouples the cross-lingual nature of the data from that of the user-facing instruction, since a model that handles English glossary specifications may not equally parse Japanese or Korean ones. This setting reflects real-world global deployments, where users typically issue translation requests in their own language rather than in English.

To avoid any ambiguity about what the headline count measures: the 7,344 items are (translation problem, instruction language) instances, and this sevenfold expansion is already reflected in that number rather than being an additional multiplier

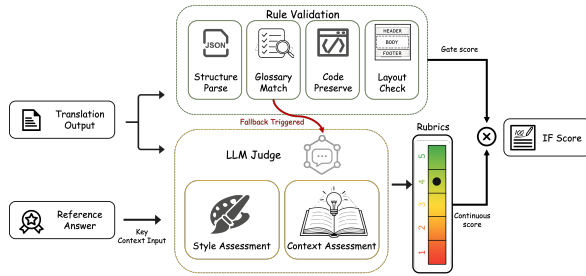


Figure 2: Hybrid evaluation pipeline. A deterministic *rule validation* module checks structure, glossary, code, and layout to produce a binary gate score. In parallel, a *rubric-conditioned LLM judge* scores style and context on a 0–5 scale, normalized to $[0, 1]$. The gating product and continuous mean are multiplied to yield the final *IF_Score*.

on top of it. The counts therefore should not be read as 7,344 independent source texts. The repetition is deliberate rather than inflated bookkeeping. Attributing a cross-language gap to instruction language requires holding the translation problem—source, target, constraint set, and gold reference—fixed while varying only the instruction language; without this matched-pair structure, any observed gap could instead reflect differences in source difficulty, target language, or constraint type. The per-instruction-language columns of Table 3 are directly comparable precisely because of it. The corpus is not multi-parallel in the usual sense: each source text is tied to a single source-target direction and is not reused across language pairs.

Balance. We control coverage across constraint type, translation direction, and instruction language so that each combination is supported by a statistically meaningful number of items while the overall distribution still reflects realistic demand.

5 Evaluation Framework

Building on the taxonomy of Section 3, IFMT-Bench adopts a hybrid evaluation pipeline that combines deterministic rule-based checks for gating constraints with a rubric-conditioned LLM judge for continuous ones, and reports both the prompt-level *IF_Score* and the translation-quality score xCOMET-XXL as two decoupled axes. Figure 2 illustrates the pipeline.

5.1 Rule-Based Checkers for Gating Constraints

For glossary adherence, we apply regex-based matching to verify whether every required term appears in the output with the correct sense, and a single miss zeroes the gating score for the item. Because glossary entries are typically given in ci-

tation form while morphologically rich target languages may require inflected variants, we add a fallback LLM judge that re-examines items zeroed by the regex stage, conditioned on the instruction, the model output, and the reference translation, to avoid penalizing legitimate morphological adaptations such as pluralization or case declension.

For structured-data constraints, we run a dedicated parser to verify JSON, HTML, CSV, or Markdown well-formedness, then compare the topology of the parsed output against the source (e.g., nested key paths in JSON, tag nesting in HTML). Any parsing failure or topology mismatch zeroes the constraint. Layout, code-keep, and code-tag constraints are checked by string-level verifiers that ensure protected separators, inline code spans, and code-tag markers are preserved verbatim in the output.

5.2 Rubric-Based Judge for Continuous Constraints

Style alignment and context-conditioned translation are scored jointly by a single LLM judge call that returns two 0-to-5 rubric scores, one per dimension, normalized to $[0, 1]$. The judge is an open-weight reasoning model (gpt-oss-120b) decoded at temperature 0, so that scoring is reproducible without depending on a proprietary API. Following Kocmi and Federmann (2023) and rubric-style LLM-as-a-judge protocols, the prompt enumerates explicit criteria for each rubric level; the complete rubric and judge prompt templates are given in Appendix A. To guard against latent miscategorization by the judge, the prompt additionally requires the judge to identify which constraint dimension each rubric addresses, and we discard the score when this auxiliary classification disagrees with the ground-truth label of the item. This sanity check reduces judge variance without retraining and yields the consistency we observe across runs.

5.3 Aggregation and Two-Dimensional Reporting

The prompt-level *IF_Score* aggregates per-constraint scores via Equation 1. The multiplicative gating term reflects the engineering intuition that a structurally broken translation is unusable regardless of stylistic polish. The per-model score is the arithmetic mean of *IF_Score* across all evaluated items.

Independently, we score translation quality with xCOMET-XXL (Guerreiro et al., 2024), which cur-

rently exhibits the strongest correlation with human judgments among reference-based metrics and natively supports all seven target languages. Reporting *IF_Score* and xCOMET-XXL as two decoupled axes enables direct identification of regimes where a model trades constraint adherence for fluency or vice versa, as analyzed in Section 6.7.

Cross-lingual applicability. The rule-based checkers are language-agnostic by construction, since JSON well-formedness and HTML tag balance do not depend on whether the target language is Chinese or German. The LLM judge is conditioned on the target-language identifier, so its rubric is interpreted in the correct linguistic context. xCOMET-XXL natively covers all seven languages. Together, these design choices give IFMTBench a uniform evaluation surface across its full language coverage without requiring per-language calibration.

6 Experiments

6.1 Models

We evaluate 15 models drawn from three camps, namely a closed-source commercial system, open-source general-purpose LLMs, and open-source translation-specialized models. The closed-source system is Gemini 3.1 Pro (Gemini Team, Google, 2024). The open-source general models include the Qwen3.5 family (0.8B, 2B, 4B, 9B, A3B, 27B) and Qwen3.6 A3B (Yang et al., 2025), as well as the gemma4 family (E2B, E4B, A4B, 31B). The translation-specialized models are from the Hy-MT2 family (Zheng et al., 2026), denoted Hy-MT2 1.8B (Dense), Hy-MT2 7B (Dense), and Hy-MT2 A3B (MoE, around 3B active out of 30B), which are post-trained with translation-focused RL recipes. This selection covers parameter scales from 0.8B to 31B and includes both dense and mixture-of-experts architectures.

6.2 Configuration

All open-source models are run with their officially recommended decoding settings and a maximum generation length of 4096 tokens with non-think model. The LLM judge is gpt-oss-120b at temperature 0, as specified in Section 5.2. The evaluation set comprises 4,506 single-constraint and 2,838 multi-constraint items, for a total of 7,344.

6.3 Overall Results

Table 2 reports *IF_Score* and xCOMET-XXL for each model under both single- and multi-constraint conditions, alongside an aggregate *IF_Score*_{total} that weights the two subsets by item count. Figure 3 visualizes the same data as a model-by-model dumbbell plot that exposes the single-to-multi gap at a glance.

Two trends are immediate. First, instruction following capability scales sharply with model size. Along the Qwen3.5 ladder, the single-constraint *IF_Score* rises monotonically from 40.92 (0.8B) to 82.48 (27B), and the multi-constraint score rises from 7.46 to 78.81, a more than tenfold increase. Models below the 2B parameters essentially fail under multi-constraint composition. Second, translation-specialized post-training narrows the gap with closed-source systems on single-constraint items, and Hy-MT2 A3B reaches 91.94, on par with Gemini 3.1 Pro at 91.95. Under multi-constraint composition, however, Gemini 3.1 Pro retains a clear lead of around 8.7 points over Hy-MT2 A3B with a multi-constraint degradation rate of only 8.1%, against 17.6% for Hy-MT2 A3B. The transition from parity to a clear gap suggests that the remaining advantage of frontier closed-source systems lies in their ability to track multiple heterogeneous constraints concurrently, a property that translation-specific post-training has not yet matched.

6.4 Per-Dimension Analysis

Figure 4 reports per-dimension *IF_Score* as paired heatmaps: panel (a) covers all seven single-constraint dimensions; panel (b) is restricted to the four dimensions with sufficient multi-constraint co-occurrence.

A clear difficulty gradient emerges. Layout and code-keep constraints are near-saturated (most models > 0.90), while glossary, code-tag, and context exhibit much wider cross-model spreads (0.71, 0.66, 0.38). Even Gemini 3.1 Pro falls short of a perfect glossary score, indicating that targeted lexical control remains non-trivial for current decoders.

Under composition, per-constraint difficulty is deliberately moderated: generating coherent texts satisfying multiple non-conflicting constraints becomes intractable at high per-constraint intensity. Despite this, the structure dimension drops from above 0.94 to between 0.55 and 0.72—even Gem-

Model	IFMTBench (ours)					Align.	General IF	
	S-IF	S-xC	M-IF	M-xC	IF _T	tax	IFE	IFB
Hy-MT2 1.8B	76.76	80.74	57.61	67.80	69.36	+3.98	80.22	35.33
Hy-MT2 7B	89.73	84.29	72.67	72.76	83.14	−5.44	86.14	35.33
Hy-MT2 A3B	91.94	85.55	75.80	74.79	85.70	−6.39	89.80	50.67
gemma4 E2B	63.71	80.22	50.72	66.92	58.69	+16.51	80.44	26.00
gemma4 E4B	74.64	82.26	69.25	67.77	72.56	+7.62	85.76	32.00
gemma4 A4B	83.17	83.30	73.89	69.61	79.58	+0.13	89.80	45.60
gemma4 31B	84.00	83.56	72.64	69.50	79.61	−0.44	93.16	48.67
Qwen3.5 0.8B	40.92	70.21	7.46	52.74	27.99	+29.29	55.08	19.67
Qwen3.5 2B	54.77	75.77	24.60	61.83	43.11	+21.00	67.10	25.33
Qwen3.5 4B	72.58	80.16	57.23	66.44	66.65	+7.58	81.15	27.33
Qwen3.5 9B	75.59	80.80	64.28	67.62	71.22	+5.21	84.29	37.00
Qwen3.5 27B	82.48	82.64	78.81	69.29	81.06	+0.16	86.88	43.33
Qwen3.5 A3B	76.46	81.74	70.32	68.48	74.09	+5.28	85.95	38.67
Qwen3.6 A3B	77.95	82.56	71.43	68.91	75.43	+4.61	83.00	36.00
Gemini 3.1 Pro	91.95	83.31	84.53	69.65	89.08	−8.64	96.30	71.34

Table 2: Overall results on IFMTBench and comparison with general instruction following benchmarks. S- and M- denote single- and multi-constraint subsets, IF is the *IF_Score* (%), xC is xCOMET-XXL (%), and IF_T is the item-count-weighted total *IF_Score*. **Align. tax** is the single-constraint gap S-xC − S-IF, where positive values mark “constraint-blind” models with fluent output but weak adherence, and negative values mark frontier models that trade fluency for stricter constraint following. IFE and IFB report prompt-level strict scores on IFEval and IFBench.

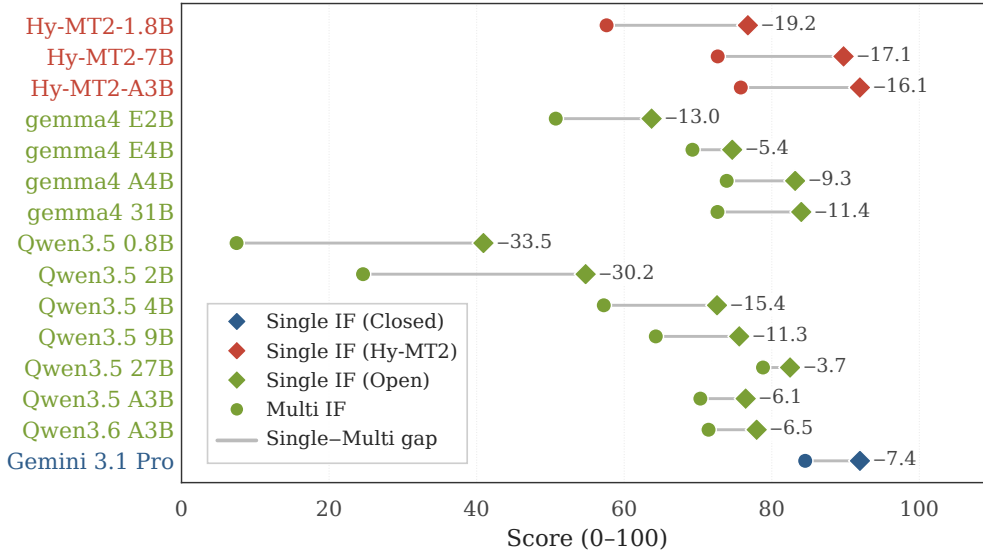


Figure 3: Overall results on IFMTBench. Each row is one of 15 models, ordered by aggregate *IF_Score*_{total}. Filled diamonds denote single-constraint *IF_Score* and filled circles denote multi-constraint *IF_Score*. Δ annotates the absolute drop under composition. Models are color-coded by family.

ini 3.1 Pro is capped at 0.72—while style, glossary, and context degrade more gracefully. This asymmetric pattern suggests that models reallocate capacity toward semantic constraints at the expense of structural ones, identifying structured-format preservation as a target for future alignment work.

6.5 Multilingual Behavior

Since IFMTBench paraphrases each task into all seven instruction languages while holding the source–target pair fixed, Table 3 directly isolates the effect of instruction language on constraint following.

Two findings emerge. First, instruction language introduces a systematic performance gap:

the average per-model range of *IF_Score* across instruction languages is 0.054 (single) and 0.050 (multi)—consistently wider than the corresponding xCOMET-XXL spread, confirming that instruction parsing generalizes less uniformly than translation quality. Second, this gap narrows sharply with scale. Qwen3.5 0.8B shows a relative range of 26.9% (amplified to 53.8% under composition), while Gemini 3.1 Pro stays within 3.3% (1.2% under composition). Sufficient multilingual exposure across pretraining and post-training appears prerequisite for instruction language robustness: parameter-poor models suffer both lower absolute scores and pronounced cross-language instability.

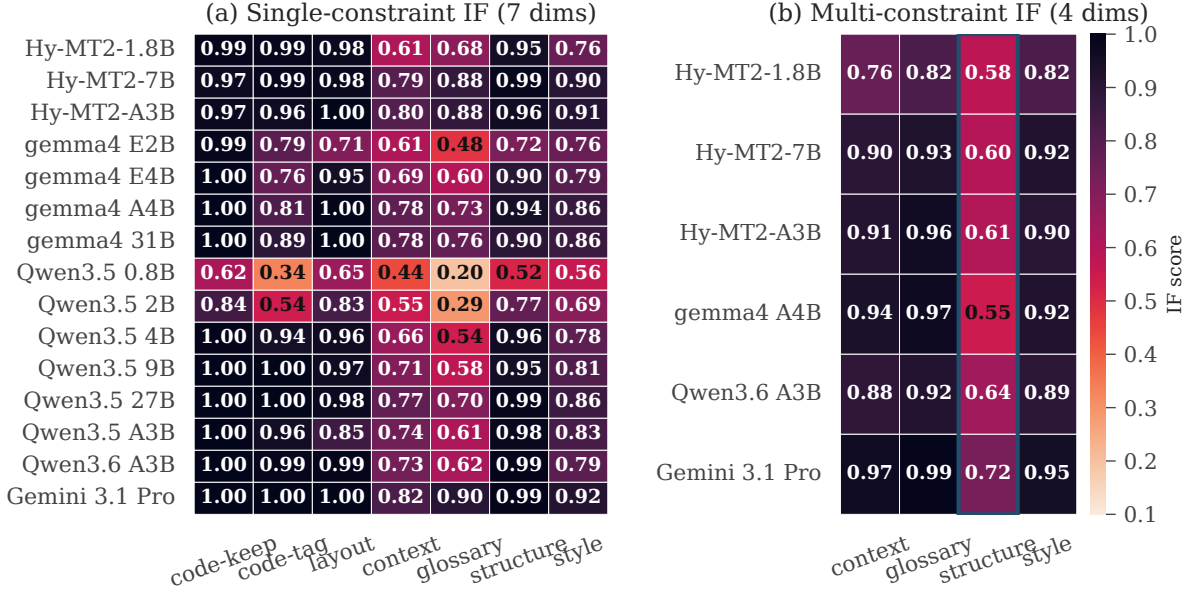


Figure 4: Per-dimension *IF_Score* heatmaps. **(a)** Single-constraint: 15 models $\times$ 7 dimensions. **(b)** Multi-constraint: 4 co-occurring dimensions. Darker cells indicate higher scores. The structure column (boxed) degrades most sharply under composition; glossary remains the most discriminative dimension.

Model	Instruction language (single / multi)							Mean
	zh	de	ja	fr	en	es	ko	
Hy-MT2 1.8B	0.84/0.63	0.75/0.56	0.72/0.54	0.77/0.59	0.81/0.62	0.75/0.58	0.73/0.52	0.77/0.58
Hy-MT2 7B	0.91/0.75	0.90/0.72	0.89/0.70	0.91/0.73	0.91/0.75	0.89/0.72	0.88/0.71	0.90/0.73
Hy-MT2 A3B	0.93/0.75	0.90/0.75	0.89/0.74	0.90/0.77	0.91/0.76	0.89/0.77	0.87/0.73	0.90/0.75
gemma4 E2B	0.67/0.54	0.62/0.48	0.64/0.52	0.61/0.50	0.66/0.53	0.63/0.50	0.63/0.47	0.64/0.51
gemma4 E4B	0.76/0.71	0.74/0.69	0.73/0.69	0.74/0.68	0.77/0.72	0.74/0.69	0.74/0.67	0.75/0.69
gemma4 A4B	0.85/0.73	0.83/0.74	0.83/0.76	0.80/0.72	0.84/0.75	0.84/0.73	0.82/0.75	0.83/0.74
gemma4 31B	0.85/0.74	0.83/0.71	0.83/0.73	0.83/0.72	0.85/0.73	0.84/0.73	0.84/0.73	0.84/0.73
Qwen3.5 0.8B	0.45/0.09	0.39/0.07	0.40/0.06	0.40/0.08	0.47/0.09	0.39/0.08	0.36/0.05	0.41/0.07
Qwen3.5 2B	0.59/0.30	0.54/0.24	0.55/0.24	0.53/0.24	0.58/0.27	0.52/0.23	0.52/0.20	0.55/0.25
Qwen3.5 4B	0.75/0.58	0.73/0.56	0.71/0.57	0.72/0.57	0.75/0.60	0.71/0.56	0.71/0.56	0.73/0.57
Qwen3.5 9B	0.77/0.65	0.76/0.63	0.75/0.63	0.75/0.66	0.78/0.65	0.74/0.66	0.74/0.62	0.76/0.64
Qwen3.5 27B	0.85/0.79	0.83/0.79	0.81/0.79	0.82/0.79	0.84/0.79	0.82/0.80	0.81/0.77	0.82/0.79
Qwen3.5 A3B	0.78/0.73	0.76/0.71	0.75/0.68	0.74/0.71	0.79/0.70	0.76/0.70	0.76/0.69	0.76/0.70
Gemini 3.1 Pro	0.92/0.85	0.92/0.85	0.92/0.84	0.90/0.84	0.93/0.85	0.93/0.85	0.91/0.85	0.92/0.85

Table 3: Per-instruction-language *IF_Score* for both **single**-constraint (black) and **multi**-constraint (gray) settings. Each column corresponds to the language in which the instruction is written, with the source–target pair held fixed. The rightmost **Mean** column averages across instruction languages. Rows are grouped by family (Hy-MT2, gemma4, Qwen3.5) with size ascending; Gemini 3.1 Pro is shown last as the closed-source reference.

6.6 Single- vs Multi-Constraint Degradation

To isolate the effect of constraint composition, Figure 5 contrasts single- and multi-constraint *IF_Score* for six representative models and decomposes the drop along the four co-occurring dimensions.

The degradation rate is strongly anti-correlated with baseline capability, spanning nearly an order of magnitude from 8.1% (Gemini 3.1 Pro) to 81.8% (Qwen3.5 0.8B). Constraint composition does not impose a uniform difficulty premium; it amplifies underlying weaknesses. When a model already struggles to obey a single constraint reliably, the multiplicative gating rule turns small per-constraint slips into total failures. Conversely, Gemini 3.1 Pro keeps its semantic constraints almost intact under

composition and loses ground mainly on structure, and because only one gating factor degrades, the multiplicative penalty stays bounded. This matches the design intent of Equation 1, namely that the metric should be lenient on continuous degradation but severe on hard-constraint failure.

6.7 Decoupling Quality from Instruction Following

The two-axis evaluation also exposes a non-linear relationship between translation quality and instruction following. The *alignment tax* column in Table 2, defined as $S\text{-}xC - S\text{-}IF$, makes this gap directly readable per model.

Positive alignment-tax values dominate the small-model regime. Qwen3.5 0.8B exhibits a 29.29-point gap, indicating that its outputs are es-

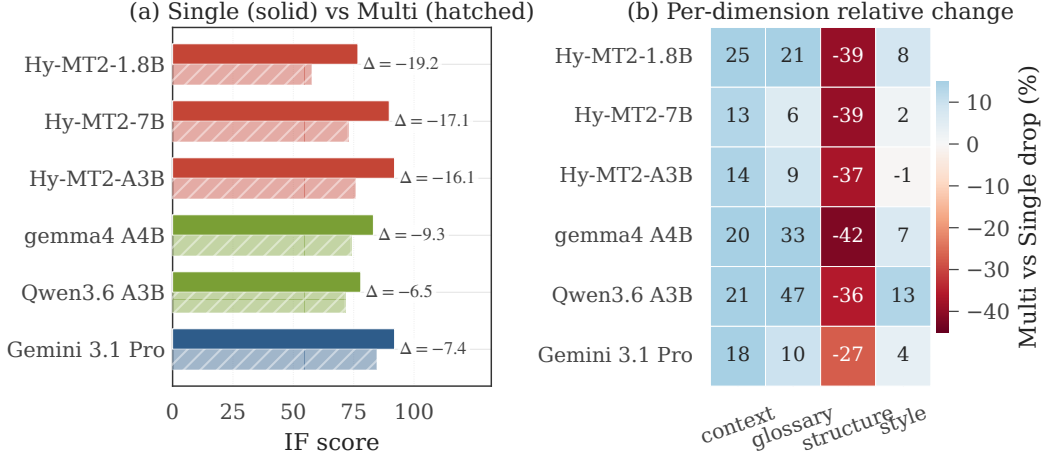


Figure 5: Degradation under constraint composition. **(a)** Single- versus multi-constraint IF_Score for six representative models. The hatched bar is the multi-constraint score and $-\Delta$ annotates the absolute drop. **(b)** Relative drop (%) per dimension. Structure is uniformly the deepest red column, suggesting that structural constraints absorb most of the composition penalty.

entially fluent free translations that largely ignore the auxiliary instructions. As model capability grows, the gap narrows and reverses sign. Hy-MT2 A3B and Gemini 3.1 Pro post negative taxes of -6.39 and -8.64 , sometimes sacrificing surface fluency to obey constraints. This instruction-first behavior is precisely what industrial workflows expect.

The decoupling has direct implications for evaluation. Within the Qwen3.5 series, xCOMET-XXL rises from 70.21 to 80.16 between 0.8B and 4B (a 10-point gain that a quality-only protocol would call moderate), while IF_Score jumps from 40.92 to 72.58 (a 32-point shift reflecting a qualitative leap in constraint following). The amplification factor suggests that a quality metric alone is a poor proxy for instruction adherence.

6.8 Comparison with General Instruction Following Benchmarks

A natural objection is that general instruction following benchmarks such as IFEval and IFBench may already approximate translation instruction following. We test this hypothesis by running the same 15 models on both benchmarks and comparing their prompt-level strict scores against IFMT-Bench (numbers in Table 2, columns IFE and IFB).

Aggregated across all 15 models, the Spearman rank correlation between IFEval and IFMT-Bench IF_Score is $\rho = 0.92$, and IFBench gives $\rho = 0.87$. These correlations look strong but are dominated by parameter size, since the smallest models sit at the bottom of every leaderboard and the largest at the top, masking finer-grained ordering. Restricting to the top 8 models on IFMT-Bench IF_Score , the correlations drop to $\rho = 0.65$

($\tau = 0.47$, $p = 0.105$) for IFEval and $\rho = 0.55$ ($\tau = 0.43$, $p = 0.179$) for IFBench. Concrete rank swaps illustrate the divergence. Among the open-weight models, gemma4 31B leads IFEval at 93.16 but falls to fourth on IFMTBench, while Hy-MT2 7B rises from fifth on IFEval to second on IFMTBench. xCOMET-XXL rules out a quality explanation, since the two score 83.56 vs 84.29 on single-constraint xCOMET-XXL yet 79.61 vs 83.14 on IF_Score_T . In the regime where ranking matters, general instruction following is no longer a reliable predictor of translation instruction following.

7 Conclusion

We presented IFMTBench, a benchmark for multilingual translation instruction following. IFMT-Bench formalizes translation constraints into gating and continuous subsets, defines seven dimensions and five compositional patterns from industrial workflows, and aggregates per-constraint scores multiplicatively. It contains 4,506 single- and 2,838 multi-constraint items across seven languages, evaluated with a hybrid rule-plus-judge protocol reporting IF_Score alongside xCOMET-XXL. Evaluating 15 frontier and translation-specialized models exposes that instruction following scales much more sharply than translation quality, that structural constraints absorb most of the composition penalty, and that general instruction following rankings cease to predict translation behavior in the regime where ranking actually matters.

Limitations

IFMTBench currently covers seven high-volume languages, leaving out scripts with distinctive properties such as Arabic or Hindi, and future iterations should extend toward lower-resource and morphologically richer targets. The seven constraint dimensions cover the most common workflow requirements but do not yet include length control, forbidden vocabulary, or fine-grained polysemy beyond glossary entries. The dataset is also static, which may erode discriminative power as models improve, so an adaptive variant that synthesizes harder items in response to current frontier behavior is a natural next step. Finally, IFMTBench’s scoring rule is directly compatible with reward modeling for GRPO-style post-training, and the early signs from translation-specialized open-source models suggest that closing the evaluation-training loop is a promising direction for future work.

Ethics Statement

IFMTBench is built through synthetic generation followed by professional human review. All source segments, references, glossaries, and background contexts are produced by the synthesis pipeline rather than scraped from user-generated content, which avoids re-distributing copyrighted material from the open web. The human verification stage explicitly includes a safety pass that removes sensitive content and de-identifies any synthetic names, addresses, or contact numbers that may surface in domain-realistic prompts (for example marketing copy or game dialogues). Annotators were professional translators working under standard hourly contracts in their respective regions, with task instructions delivered in their native language and no exposure to harmful content beyond what is normal for commercial translation review. The benchmark is intended for evaluation rather than training, and the LLM judge used in our protocol is a general-purpose open-source model accessed through its public API. We release IFMTBench under a license that permits non-commercial research use, with redistribution of the underlying source segments restricted to avoid memorization-based contamination of future translation models.

References

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural ma-

chine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3063–3068.

Gemini Team, Google. 2024. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Nuno M Guerreiro, Ricardo Rei, Daan van Stigt, Luísa Coheur, Pierre Colombo, and André F T Martins. 2024. xCOMET: Transparent machine translation evaluation through fine-grained error detection. *Transactions of the Association for Computational Linguistics*, 12:979–995.

Yun He, Di Jin, Chaoqi Wang, Chloe Bi, Karishma Mandyam, Hejia Zhang, Chen Zhu, Ning Li, Tengyu Xu, Hongjiang Lv, Shruti Bhosale, Chenguang Zhu, Le Hou, Yue Wang, Hanchao Mao, Sergey Edunov, Aman Mahajan, Boyu Xue, Tianlu Liu, and 3 others. 2024. MultiIF: Benchmarking LLMs on multi-turn and multilingual instructions following. *arXiv preprint arXiv:2410.15553*.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. 2024. Follow-Bench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Tom Kocmi, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Philipp Koehn, and 1 others. 2023. Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In *Proceedings of the Eighth Conference on Machine Translation (WMT)*.

Tom Kocmi and Christian Federmann. 2023. Large language models are state-of-the-art evaluators of translation quality. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 193–203.

Llama Team, AI @ Meta. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems (NeurIPS)*.

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

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation (WMT)*, pages 392–395.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. 2025. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. InFoBench: Evaluating instruction following ability in large language models. *arXiv preprint arXiv:2401.03601*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems (NeurIPS)*.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y K Li, Y Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

An Yang and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Mao Zheng, Zheng Li, Tao Chen, Bo Lv, Mingrui Sun, Mingyang Song, Jinlong Song, Hong Huang, Decheng Wu, Hai Wang, Yifan Song, Yanfeng Chen, and Guanwei Zhang. 2026. [Hy-mt2: A family of fast, efficient and powerful multilingual translation models in the wild](#). *Preprint*, arXiv:2605.22064.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Sid-dhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

A LLM Judge Prompt Templates

This appendix provides the full prompt templates used in our hybrid evaluation framework (Section 5). We include both the glossary fallback judge prompt (Section A.1) and the style/background rubric judge prompt (Section A.2).

A.1 Glossary Fallback Judge Prompt

The glossary fallback judge is invoked when the regex-based validator returns a zero score, to account for legitimate morphological adaptations in

the target language. It produces a binary 0/1 decision. The full prompt template is shown below.

System Prompt: Glossary Compliance Judge

ROLE

You are an expert Linguistic and Morphological Evaluator for a translation Reward Model. Your SOLE objective is to determine if the specified terminology from the instruction was correctly integrated into the target translation, accounting for complex morphological adaptations (declension, pluralization, tense, etc.).

EVALUATION DATA

```
<instruction> {user_instruction} </instruction>
<ground_truth> {ground_truth} </ground_truth>
<model_output> {target_translation}
</model_output>
```

RUBRICS

Glossary Compliance - [BINARY SCORING: 0 or 1]

Evaluate if the translation accurately incorporates the specific terminology provided in the instruction/background.

- **[1] Perfect Adherence:** Flawlessly integrated the required terms. Morphological adaptations (e.g., plurals, tense, conjugations) are grammatically natural in the target language. It is acceptable if the term underwent necessary morphological changes compared to its base dictionary form.

- **[0] Fatal Violation (Veto):** Instant 0 if ANY of the following occur: unauthorized synonym substitution, fallback to generic dictionary translation, omission of the core concept, or severe grammatical corruption caused by forcing the term.

OUTPUT FORMAT

Output ONLY a single integer: 1 or 0. Do NOT wrap it in JSON, Markdown, or any other formatting. Do NOT output any explanatory text.

A.2 Style and Background Judge Prompt

The style/background judge scores two continuous constraints simultaneously on a 0-to-5 rubric, normalized to [0, 1]. The prompt includes an activation condition for each rubric dimension: if the instruction does not request a particular constraint, the judge outputs null for that dimension. The full prompt template is shown below.

System Prompt: Style & Background Rubric Judge

ROLE

You are an advanced Reward Model designed for Reinforcement Learning (RL) of Large Language Models. Your primary function is to evaluate **Instruction Tracking and Constraint Satisfaction**. Do NOT evaluate basic translation fluency. Your SOLE objective is to score whether the model executed the specific holistic [Constraints] (Style and Background).

EVALUATION DATA

```
<instruction> {user_instruction} </instruction>
<ground_truth> {ground_truth} </ground_truth>
<model_output> {target_translation} </model_output>
```

RUBRICS

Analyze the <instruction>. If a constraint is NOT requested, output null. If activated, evaluate against the rubrics below.

1. Style & Register - [0-5 SCALE]

Activation Condition: Activate if the instruction requests a specific tone, persona, register, or formatting style.

- [5] Perfect Alignment: Tone and register are exceptionally distinct and consistent throughout.
- [4] Strong Alignment: Generally fits the required style, but 1–2 lexical choices feel slightly generic.
- [3] Marginal Pass: Follows the basic directional constraint, but leans heavily on standard, flavorless translation.
- [2] Default/Generic: Ignored the stylistic constraint, reverting to a safe, bland machine translation tone.
- [1] Severe Deviation: Noticeable conflict with the requested style.
- [0] Rule Break: Wrong style AND included conversational filler/hallucinations, breaking the fourth wall.

2. Contextual Cohesion (Background) - [0-5 SCALE]

Activation Condition: Activate if the instruction provides ANY preceding context, a background summary, or asks the translation to consider the “context” or “background”.

- [5] Perfect Disambiguation: Masterfully leveraged the background summary to resolve potential ambiguities. Flawless logical cohesion.
- [4] Strong Utilization: Correctly used the summary to guide the translation, but feels slightly rigid when referencing the background.
- [3] Logically Consistent: Does not contradict the summary, but disambiguation is mediocre (literal translation).
- [2] Total Ignorance: Ignored the summary entirely, resulting in a disjointed literal translation.
- [1] Logical Contradiction: Directly contradicts the core logic or established facts in the background summary.
- [0] Severe Hallucination (Prompt Bleeding): Mistakenly translated the background summary itself as part of the target text.

OUTPUT FORMAT

Output ONLY a valid JSON object. Do NOT wrap the JSON in Markdown code blocks.

```
{"scores": {"style": [0-5 or null], "background": [0-5 or null]}}
```
