LLM-as-a-Jury for Machine Translation Publishability Assessment

Olivia Norris and **Alex Yanishevsky**
Smartling
244 Fifth Avenue Suite 1471 New York, NY 10001
{onorris,ayanishevsky}@smartling.com

Abstract

We propose an LLM-as-a-Jury framework for determining machine translation publishability, aggregating judgments from multiple large language models via logistic regression rather than relying on a single judge. Publishability is defined as the absence of major or critical errors—those that render a translation unsuitable for public release without human post-editing. We compare three evaluation frameworks: a generic Edit Effort Estimation (EEE) prompt based on lexical accuracy, grammatical correctness and semantic coherence, a generic Linguistic Quality Assurance (LQA) prompt based on the MQM error taxonomy, and a purpose-built Publishability prompt optimized via DSPy and augmented with domain-specific fine-tuning. Experiments across three domains and nine language pairs show that (i) the jury ensemble matches or outperforms the best individual juror in nearly every condition, (ii) EEE and LQA juries are competitive with and occasionally exceed the Publishability jury on macro-F1, (iii) the Publishability framework offers stronger precision and a more favorable error correction asymmetry, and (iv) domain-specific fine-tuning yields substantial recall gains in client-heavy domains. These results support the viability of fully automated publishability determination in enterprise MT workflows.

1 Introduction

Large language models are increasingly used as automated evaluators of machine translation quality (Kocmi and Federmann, 2023b; Fernandes et al., 2023), a paradigm known as "LLM-as-a-Judge" (Zheng et al., 2023). While single-judge approaches demonstrate strong system-level correlation with human assessments, they are susceptible to intra-model bias and offer limited robustness guarantees for high-stakes decisions (Verga et al., 2024).

This work addresses a commercially significant use case: raw AI translation, in which content is published with no human post-editing (also referred to as no HITL or no human-in-the-loop). The core decision is binary—does a given translation contain an error severe enough to warrant human review before publication? We define a *publishable* translation as one free of major or critical errors per the MQM framework (Freitag et al., 2021), deliberately including translations with minor or neutral errors to reflect the operational threshold for public release rather than a zero-defect standard.

Rather than relying on a single LLM judge, we propose an **LLM-as-a-Jury** approach in which multiple LLMs from diverse model families independently evaluate each segment and their outputs are aggregated via a learned logistic regression model, building on evidence that panels of diverse models reduce correlated bias and outperform single judges (Verga et al., 2024; Li et al., 2025).

Contributions We formalize MT publishability as a jury-based binary classification task augmented with client-specific constraints; introduce a multi-stage Publishability framework comprising

automated rule induction, retrieval-augmented examples, and DSPy-optimized prompting; demonstrate a consistent error correction asymmetry across all frameworks in which the jury corrects individual errors far more often than it introduces them; and show that domain-specific fine-tuning yields recall improvements of up to 43 percentage points in client-heavy domains.

2 Related Work

LLM-as-a-Judge for MT Zheng et al. (2023) showed that strong LLM judges can match inter-annotator agreement, while subsequent work identified systematic biases including position, verbosity, and self-enhancement effects. Kocmi and Federmann (2023b) introduced GEMBA, a zero-shot GPT-based MT metric, extended by Kocmi and Federmann (2023a) to GEMBA-MQM for fine-grained error span detection. Fernandes et al. (2023) showed that prompting for explicit error analysis outperforms scalar score prediction, and Lu et al. (2024) validated chain-of-thought error analysis across over 100,000 segments. G-Eval (Liu et al., 2023) demonstrated strong NLG evaluation alignment while raising the concern that LLM evaluators favor LLM-generated text.

From Single Judge to Jury Verga et al. (2024) demonstrated that aggregating judgments from diverse smaller models (PoLL) outperforms single large judges while reducing cost. Li et al. (2025) extended this with dynamic per-instance jury selection. Qian et al. (2026) proposed BT-σ, a judge-aware aggregation method that models differences in judge reliability.

Bias and Robustness. Translationese bias—where judges favor machine-translated over human-authored text—poses a risk of inflated quality assessments in jury-based systems (Zhang et al., 2025). Anonymous (2026) further show that annotator disagreement often reflects competing severity interpretations, underscoring the value of jury diversity in capturing multiple valid evaluative perspectives.

3 Methodology

3.1 Task Definition

A segment is labeled *unpublishable* if it contains at least one major or critical error, and is considered *publishable* otherwise. This includes segments with minor or neutral errors in the publishable class, reflecting an operational rather than zero-defect quality threshold. Error categories follow MQM conventions (Freitag et al., 2021), augmented with client-specific types: omission, mistranslation, formality violation, glossary violation, and brand substitution.

3.2 Data Curation

Data were drawn from the company's internal translation database. Three domains with sufficient linguist-generated LQA annotations were identified—Logistics, IT, and Media/Entertainment—and all language pairs with at least 1,000 labeled segments were retained, yielding nine locale datasets (Table 1). The retained language pairs span a typologically diverse set of languages, including European languages (German, Greek, French, Spanish, Italian, Portuguese), a morphologically complex agglutinative language (Indonesian), and a logographic language with distinct syntactic structure (Japanese), covering both Latin and non-Latin scripts. Severity annotations (Minor, Neutral, Major, Critical) assigned by professional linguists were used to derive the binary publishability label. No inter-annotator agreement statistics were computed; in this commercial context, linguist annotations function as the operational ground truth that defines the task rather than as estimates of an underlying latent quality variable. The binary reduction to major/critical versus all other severities targets the boundary where MQM annotator disagreement is known to be lowest, further limiting the practical impact of label noise.

Each dataset was partitioned into four non-overlapping splits: a training set and validation set for rule induction, prompt optimization, and fine-tuning; a logistic regression (LR) training set held out from all of those steps and used exclusively to fit the jury aggregator; and a test set withheld from all training for final evaluation. Sample selection relied on convenience sampling based on available linguist-reviewed data, limiting the generalizability of conclusions to the specific domains and language pairs studied.

3.3 Evaluation Frameworks

Edit Effort Estimation (EEE) An existing production prompt instructs the model to enumerate errors related to semantic coherence, grammatical correctness, and lexical accuracy, producing an aggregate quality label of *low*, *medium*, or *high* ex-

Table 1: Dataset splits by domain and target locale (source: en-US).

Domain	Locale	Train	Valid	LR Train	Test
Logistics	de-DE	39	10	40	60
Logistics	el-GR	94	24	40	60
Logistics	fr-FR	167	41	43	60
IT	es-ES	353	88	142	61
IT	id-ID	408	102	162	70
IT	it-IT	182	46	51	60
IT	pt-BR	905	226	348	149
Media	es-US	277	69	101	60
Media	ja-JP	688	171	267	115

pected post-editing effort. Each segment is enriched with the retrieved examples of a golden standard translation using Elasticsearch from the linguist reviewed translation memory units in our platform. Mapping this ordinal output to a binary publishability decision requires a choice about the *medium* class; both conventions are evaluated and reported separately. Given our publishability definition, we expect the individual jurors on the medium = publishable convention to perform better.

Linguistic Quality Assurance (LQA) An existing production prompt instructs the model to identify errors across standard MQM dimensions—accuracy, fluency, terminology, and style—and produce an error severity label for each identified error. The LQA output captures each identified error and its associated severity level (minor, neutral, major, or critical); if any error of major or critical severity is present, the segment is mapped to unpublishable (0), and publishable (1) otherwise. Each segment is enriched with the same glossary terms based on lemmatization and examples of a golden standard translation using Elasticsearch from the linguist reviewed translation memory units in our platform.

Publishability Framework A purpose-built evaluation pipeline designed specifically for binary critical and major-error detection. Each segment is enriched with three additional context fields before inference: (1) *glossary terms*, extracted from client glossaries and matched to source text via a preprocessing script that employs lemmatization techniques; (2) *retrieved examples*, comprising the five most similar publishable and five most similar unpublishable segments from the training set, retrieved using BM25 lexical similarity (implemented via Elasticsearch); and

Table 2: DSPy optimizer comparison by macro-F1, request volume, and cost.

Optimizer	Macro-F1	LLM Requests	Cost (USD)
MIPROv2	0.42	2,180	$48.05
GEPA	0.46	3,915	$89.34
SIMBA	0.48	7,650	$161.77

(3) *induced rules*, a structured set of domain- and locale-specific publishability heuristics generated by prompting GPT-4o to identify recurring patterns in the training data, applied independently per domain–locale pair.

The Publishability prompt was optimized using DSPy (Khattab et al., 2023), with three optimizers evaluated on Gemini 2.5 Pro: MIPROv2, GEPA, and SIMBA. SIMBA achieved the highest macro-F1 and its resulting prompt was adopted for all subsequent inference (Table 2). The same prompt was applied across all juror models without per-model re-optimization; as optimization was performed on Gemini 2.5 Pro, Gemini-family models may hold a latent performance advantage in individual juror comparisons. However, individual juror rankings are not the paper's primary focus — the central question is whether the jury ensemble outperforms its best individual member, irrespective of which model that is. DSPy optimization is used here as a principled alternative to manual prompt engineering rather than as a mechanism for model comparison, and the jury's error correction asymmetry holds regardless of which juror is designated as the lead.

For the fine-tuned variant, one GPT-4o and one Gemini 2.5 Pro model were fine-tuned per domain on the combined training and validation sets using the SIMBA-optimized prompt. Each fine-tuned model is multilingual within its domain. Due to token length constraints, the retrieved examples and induced rules fields were omitted during fine-tuning, which may have limited those models' ability to leverage these enrichment fields at inference time.

3.4 Jury Architecture

EEE and LQA use three jurors drawn from distinct model families: GPT-5.2, Gemini 3.1 Pro Preview, and Claude Opus 4.5. The Publishability framework expands the jury to five by adding fine-tuned GPT-4o and fine-tuned Gemini 2.5 Pro and upgrading GPT-5.2 to GPT-5.4.

The non-fine-tuned Publishability configuration

— comprising GPT-5.4, Opus 4.5, and Gemini 3.1 — constitutes a controlled 3-juror ablation directly comparable to the EEE and LQA configurations; cross-framework comparisons at equal jury size should be drawn from these rows. The fine-tuned rows reflect an extended configuration available only within the Publishability framework, as fine-tuning was performed exclusively on the Publishability prompt architecture.

Jury outputs are aggregated via logistic regression. For the Publishability framework, juror labels are binary (publishable $\rightarrow$ 1; unpublishable $\rightarrow$ 0). For LQA, juror outputs contain the identified errors and their associated severity labels; following the same mapping used during dataset construction, a segment is labeled unpublishable (0) if any error of major or critical severity is present, and publishable (1) otherwise. For EEE, juror outputs are first mapped to an ordinal scale (low $\rightarrow$ 0, medium $\rightarrow$ 1, high $\rightarrow$ 2) before being passed to the logistic regression model. Because higher expected effort corresponds to lower publishability, the learned weights for EEE predictors are negative: a higher ordinal value suppresses the publishability probability. The LR model then produces a publishability probability:

$$\hat{p} = \sigma\Big(\mathbf{w}^\top \mathbf{x} + b\Big), \tag{1}$$

where $\mathbf{w}$ are learned per-juror weights, b is a bias term, and σ is the sigmoid function. A binary decision is obtained via a threshold τ, selected to maximize macro-F1 on the LR training set using a fine-grained grid search over the range $(0.001, 0.999)$ with iterative refinement. The LR model is fit exclusively on the held-out LR training set using scikit-learn's `LogisticRegression` with `max_iter=1000`.

4 Results

All results are evaluated on the held-out test set using macro-F1, precision, and recall. We additionally report two jury correction statistics: % J1$\rightarrow$J, the fraction of segments where the best individual juror (Juror #1) was incorrect but the jury was correct, and % J$\rightarrow$J1, the reverse. Fitted LR coefficients and thresholds are provided in Table 3.

4.1 EEE

As expected, the individual jurors in the medium = publishable convention outperform medium = unpublishable across all domains (Table 4). Under the publishable convention, EEE jury F1 (0.478–0.582) is competitive with the best-performing individual juror in all domains, and reaches the highest jury F1 of any framework in IT (0.582). The error correction asymmetry is modest under this convention but becomes highly pronounced under medium = unpublishable, where the jury corrects the lead juror's errors in 37.5–49.4% of cases while introducing errors in only 6.8–12.2%.

4.2 LQA

LQA individual juror F1 scores are tightly clustered (0.514–0.580), reflecting greater cross-model consistency than either EEE or Publishability. The LQA jury achieves F1 of 0.556–0.582, matching or exceeding the EEE jury under the publishable convention and outperforming the non-fine-tuned Publishability jury in all three domains (Table 4). The error correction asymmetry is favorable but narrow (1.1–5.9% corrected vs. 2.1–2.9% introduced), indicating that individual LQA jurors already agree on most difficult cases. This high cross-model consistency likely reflects the fact that the publishability ground truth is itself derived from linguist-reviewed LQA annotations: segments labeled unpublishable are precisely those that received a Major or Critical severity rating in the underlying LQA workflow, making LQA-style prompts naturally well-aligned with the classification objective.

4.3 Publishability

Without fine-tuning, Publishability jury F1 (0.463–0.552) falls below the EEE (medium = publishable) juries across two domains and the LQA juries across all domains (Table 5). However, the framework achieves comparatively high precision (0.614–0.861) and the strongest error correction asymmetry of any framework, with 7.6–19.6% of lead juror errors corrected against only 2.2–2.8% introduced. The addition of fine-tuned jurors produces meaningful recall gains in Logistics (+20.3 pp) and Media (+23.5 pp), with jury F1 rising to 0.552–0.577 and approaching LQA-level performance in those domains (Table 5). The IT domain is largely unaffected by fine-tuning (F1: 0.461 vs. 0.463).

Table 3: LR coefficients and thresholds by domain and framework.

Domain	Framework	GPT-5.2	Opus-4.5	Gem-3.1	ft-GPT-4o	ft-Gem-2.5	Intercept	τ
Logistics	EEE	−0.189	−0.144	−0.495	–	–	2.213	0.680
	LQA	0.134	0.691	0.109	–	–	0.629	0.656
	Publishability (no ft)	0.172	0.285	0.535	–	–	0.872	0.708
	Publishability (ft)	0.180	0.456	0.656	0.002	0.476	0.681	0.668
IT	EEE	−0.010	−0.676	−0.002	–	–	2.510	0.760
	LQA	0.140	0.403	0.254	–	–	1.241	0.800
	Publishability (no ft)	−1.285	0.224	0.265	–	–	1.625	0.836
	Publishability (ft)	−1.294	0.225	0.277	0.042	1.115	0.467	0.836
Media	EEE	−0.445	−0.450	0.785	–	–	2.233	0.776
	LQA	−0.383	0.867	0.130	–	–	1.102	0.832
	Publishability (no ft)	0.000	0.150	−0.148	–	–	1.644	0.820
	Publishability (ft)	0.000	0.151	−0.150	−0.001	−0.147	1.786	0.820

Table 4: EEE and LQA results. EEE is reported under both medium-label conventions.

Framework	Domain	GPT-5.2	Opus-4.5	Gem-3.1	Jury F1	Prec.	Rec.	% J1→J	% J→J1
EEE (med=pub)	Logistics	0.570	0.535	0.569	0.582	0.814	0.814	12.2%	7.2%
	IT	0.412	0.582	0.515	0.582	0.880	0.904	0.0%	0.0%
	Media	0.395	0.508	0.517	0.478	0.833	0.811	8.0%	17.6%
EEE (med=unpub)	Logistics	0.372	0.335	0.361	0.582	0.814	0.814	46.1%	12.2%
	IT	0.253	0.362	0.365	0.582	0.880	0.904	49.4%	6.8%
	Media	0.272	0.398	0.398	0.478	0.833	0.811	37.5%	9.1%
LQA	Logistics	0.568	0.547	0.580	0.568	0.812	0.771	5.0%	2.8%
	IT	0.526	0.542	0.573	0.556	0.874	0.880	5.9%	2.1%
	Media	0.514	0.571	0.536	0.582	0.868	0.850	1.1%	2.9%

5 Discussion

5.1 Framework Comparison

Contrary to our initial hypothesis, the purpose-built Publishability framework does not achieve the highest macro-F1 across domains. The appropriate controlled comparison is the non-fine-tuned Publishability jury (3 jurors, Table 5) against the EEE and LQA juries. The LQA jury is competitive with or exceeds the non-fine-tuned Publishability jury in all three domains, and the EEE jury (medium = publishable) matches LQA performance in Logistics and IT. This is a noteworthy finding: task-specific prompt engineering does not guarantee superior F1 performance when generic prompts already produce coherent, well-calibrated juror outputs.

The Publishability framework's advantage lies instead in precision and error correction asymmetry. It consistently achieves higher precision than EEE and comparable precision to LQA, making it more conservative about flagging publishable content as problematic. Its error correction ratios are also the most favorable across all frameworks, suggesting that the enriched input representation—glossary terms, induced rules, and retrieved examples—helps jurors avoid idiosyncratic errors even when it does not consistently improve the majority outcome.

The LQA framework's strong performance in F1 is somewhat counterintuitive given that it was not designed for binary critical-error detection. A likely explanation is that the LQA prompt's comprehensive error taxonomy, while potentially overbroad, provides sufficient signal for the LR aggregator to calibrate a reliable threshold. Crucially, the ground truth labels themselves were derived from linguist-reviewed LQA annotations—segments are classified as unpublishable precisely when they received a Major or Critical severity rating in the underlying LQA workflow. This structural alignment between the LQA evaluation paradigm and the labeling process likely contributes to the tight clustering of individual juror F1 scores and the overall competitiveness of the LQA jury. This observation motivates a practical workflow recommendation: LQA may be a viable alternative publishability gate in contexts where

Table 5: Publishability results, with and without fine-tuning.

Framework	Domain	GPT-5.4	Opus-4.5	Gem-3.1	ft-GPT	ft-Gem	Jury F1	Prec.	Rec.	% J1→J	% J→J1
No fine-tuning	Logistics	0.178	0.407	0.465	–	–	0.542	0.614	0.542	13.9%	2.2%
	IT	0.124	0.309	0.463	–	–	0.463	0.850	0.682	7.6%	2.4%
	Media	0.139	0.471	0.439	–	–	0.552	0.861	0.552	19.6%	2.8%
Fine-tuned	Logistics	0.178	0.407	0.465	0.439	0.354	0.577	0.827	0.745	23.3%	4.4%
	IT	0.124	0.309	0.463	0.462	0.461	0.461	0.850	0.678	7.6%	2.6%
	Media	0.139	0.471	0.439	0.456	0.456	0.552	0.861	0.787	19.6%	2.8%

purpose-built prompt development is not feasible, though it is better suited as a downstream diagnostic tool for linguists than as a primary gating mechanism.

5.2 The Value of Jury Aggregation

Across all frameworks and domains, the jury is systematically more likely to correct an individual model's error than to override a correct judgment. This asymmetry is the strongest argument for jury-based aggregation, and it holds even in conditions where the jury's headline F1 does not exceed the best individual juror. It is most pronounced for EEE under the medium = unpublishable convention, where correction-to-introduction ratios reach 4–7×, and remains favorable for both Publishability variants (ratios of approximately 5–8×) and LQA (approximately 2×).

The logistic regression aggregation reinforces this by learning domain-dependent juror weights. The large variation in fitted coefficients across models and domains (Table 3) reflects genuine differences in per-model reliability: GPT-5.4 receives a strongly negative weight in the IT Publishability framework (-1.29), for instance, indicating that its predictions are anti-correlated with the ground truth in that domain. Majority voting would treat this juror equally with its peers; LR aggregation suppresses its influence. For EEE, all juror coefficients are negative, consistent with the ordinal input encoding: a higher effort score (0 = low, 1 = medium, 2 = high) implies lower publishability, so the logistic regression correctly learns an inverse relationship between the EEE score and the probability of publishability.

One notable exception is the IT domain under the Publishability framework, where jury F1 does not improve with fine-tuning and correction–introduction rates are nearly equal. This suggests that juror errors in specialized IT content may be more correlated across model families, possibly because technical terminology creates shared failure modes that the jury cannot resolve through aggregation alone.

5.3 Effect of Fine-Tuning

Fine-tuning yields substantial gains in Logistics and Media, where client-specific glossary requirements and brand constraints create error patterns underrepresented in base model pretraining. Jury recall increased by 20.3 and 23.5 percentage points in those domains, respectively, with a precision increase of 21.3 points in Logistics. These gains come at additional cost (see Appendix B) and require a minimum volume of labeled training data. Our results suggest that several hundred labeled segments per locale represent a practical threshold below which fine-tuning benefits are modest and potentially unstable, as evidenced by the Logistics de-DE split with only 49 combined training and validation examples.

6 Conclusion

We have presented an LLM-as-a-Jury framework for MT publishability assessment evaluated across three domains, nine language pairs, and three evaluation frameworks. Several conclusions emerge.

The jury approach is viable and robust. Across nearly all conditions, the ensemble matches or outperforms the best individual juror, and the error correction asymmetry—present across all frameworks—provides compelling evidence that aggregation over diverse model families acts as a reliable noise-reduction mechanism independent of headline F1 improvements.

Generic frameworks are stronger baselines than anticipated. The LQA jury achieves competitive or superior macro-F1 to the purpose-built Publishability framework in most conditions, and the EEE jury is similarly competitive under the appropriate medium-label convention. These results suggest that the value of the Publishability framework lies in its precision characteristics and error correction asymmetry rather than raw accuracy; LQA may

serve as a practical alternative where purpose-built development is not feasible. We recommend deploying LQA as a downstream diagnostic tool on rejected segments rather than as the primary publishability decision-maker.

Fine-tuning provides the strongest marginal gains for client-specific domains and is most beneficial when sufficient labeled data is available. IT content, being well-represented in base model pretraining, benefits least from domain adaptation.

Several limitations constrain generalizability: non-random sample selection, small locale-level test sets, potential noise in linguist-assigned reference labels, and the use of proprietary models with version-dependent behavior. Conclusions should be treated as specific to the domains and language pairs studied.

Future work should explore alternative aggregation methods such as BT-σ (Qian et al., 2026) and dynamic jury selection (Li et al., 2025), test the jury paradigm on related binary classification tasks in the MT pipeline, and evaluate performance on larger client-labeled datasets. Deployment as a live publish/review decision gate in production workflows would enable direct measurement of throughput and quality impact.

Sustainability Statement

All experiments were conducted on GPU hardware, with a total estimated energy consumption of 18.67 kWh and a carbon footprint of approximately 4.02 kg CO_2e. Carbon estimates were calculated using the Green Algorithms framework (Lannelongue et al., 2021), applying the formula:

$$\text{Carbon (gCO}_2\text{e)} = t \times P_{\text{total}} \times \text{PUE} \times \text{CI}$$

where P_{total} is computed as GPU TDP $\times$ usage factor + memory (GB) $\times$ 0.3725 W/GB = $700 \times 0.8 + 80 \times 0.3725 = 589.8$ W per GPU, with a PUE of 1.1 and a pragmatic scaling factor (PSF) of $3\times$ applied to all components.

The dominant contributor to the overall footprint was fine-tuning (6 models), which accounted for 6.00 GPU-hours (20.8% of total usage). The remaining workload components—DSPy Optimization, Rule Induction, EEE Inference, LQA Inference, and Publishability Inference—collectively comprised 22.79 GPU-hours. A full breakdown is provided in Table 6.

Table 6: Estimated energy and carbon cost by workload component.

Component	GPU-h	kWh	CO_2e (g)
DSPy Optimization	8.59	5.57	836
Rule Induction	0.04	0.02	7
EEE Inference	2.36	1.53	383
LQA Inference	2.36	1.53	383
Pub. Inf (no ft)	3.54	2.30	574
Pub. Inf (ft)	5.90	3.83	957
Fine-tuning (6 models)	6.00	3.89	876
Total	**28.79**	**18.67**	**4016**

In contextual terms, the total carbon footprint is equivalent to driving approximately 23.7 km in a European car, or 0.42% of a transatlantic flight (New York to London), and would require approximately 4.3 tree-months to sequester.

References

Anonymous. 2026. Using model disagreement to identify unstable regions in MT evaluation. Under review.

Fernandes, Patrick, Daniel Deutsch, Mara Finkelstein, Parker Riley, André F. T. Martins, Graham Neubig, Ankush Garg, Jonathan Clark, Markus Freitag, and Orhan Firat. 2023. The devil is in the errors: Leveraging large language models for fine-grained machine translation evaluation. In *Proceedings of the Eighth Conference on Machine Translation (WMT 2023)*, pages 1066–1083.

Freitag, Markus, 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.

Khattab, Omar, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2023. DSPy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*.

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

Kocmi, Tom and Christian Federmann. 2023b. 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 2023)*, pages 193–203.

Lannelongue, Loïc, Jason Grealey, and Michael Inouye. 2021. Green algorithms: Quantifying the carbon footprint of computation. *Advanced Science*, 8(12):2100707.

Li, Xinpeng, Kuan Wang, Gautam Gouda, Shubham Choudhary, Yanfu Wang, Lifu Hu, Jennifer Vaughan, and Freddy Lecue. 2025. Who judges the judge? LLM jury-on-demand: Building trustworthy LLM evaluation systems. *arXiv preprint arXiv:2512.01786*.

Liu, Yang, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG evaluation using GPT-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Lu, Qingyu, Baopu Qiu, Liang Ding, Kanjian Zhang, Tom Kocmi, and Dacheng Tao. 2024. Error analysis prompting enables human-like translation evaluation in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8801–8816.

Qian, Mengie, Guangzhi Sun, Mark Gales, and Kate Knill. 2026. Who can we trust? LLM-as-a-jury for comparative assessment. *arXiv preprint arXiv:2602.16610*.

Verga, Pat, Sebastian Hofstätter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. 2024. Replacing judges with juries: Evaluating LLM generations with a panel of diverse models. *arXiv preprint arXiv:2404.18796*.

Zhang, Hongbin, Kehai Chen, Xuefen Bai, Youcheng Pan, Yang Xiang, Jinpeng Wang, and Min Zhang. 2025. Mitigating translationese bias in multilingual LLM-as-a-judge via disentangled information bottleneck. *arXiv preprint arXiv:2603.10351*.

Zheng, Lianmin, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

A Prompt Templates

Rule Induction Prompt

The following is an excerpt of the prompt submitted to GPT-4o independently for each domain–locale pair, with the combined training and validation data for that pair provided as input. The output was a JSON array of rule objects subsequently inserted as a structured input field for all downstream Publishability inference.

```
You are a linguistic quality
analyst.  You will be given a
dataset of translation segments
...
Your task is to analyze
the dataset as a whole and
identify trends or rules that
determine whether a segment
is labeled ``publishable'' or
``unpublishable''.  Look for
patterns ...
```

SIMBA-Optimized Publishability Prompt

The following is an excerpt of the prompt produced by the SIMBA DSPy optimizer and used for all Publishability inference and fine-tuning. It extends the base task definition with three calibration instructions derived from optimizer feedback, directing the model to avoid penalizing literal but grammatically correct translations, to consider broader context before flagging omissions, and to weigh overall translation quality rather than isolated stylistic deviations.

```
Evaluate whether a machine
translation is publishable or
unpublishable in its current
state.  A translation is
'publishable' if it accurately
conveys the source meaning in
the target language, follows
any provided glossary and
publishability rules, and
requires no further editing
before being shown to end
users.  Otherwise, it is
'unpublishable'.  ...
```

B Fine-Tuning Details

One GPT-4o (gpt-4o-2024-08-06) model and one Gemini 2.5 Pro model were fine-tuned per domain—Logistics, IT, and Media/Entertainment—using the SIMBA-optimized prompt and the combined training and validation sets for all locales within that domain. Each fine-tuned model is therefore multilingual within its domain. The training examples were formatted using the SIMBA prompt structure with source locale, target locale, source text, target text, and glossary terms as inputs and the binary publishability label as the target output. The retrieved examples and induced rules fields were omitted from training due to token length constraints.

GPT-4o was fine-tuned via the OpenAI fine-tuning API with `n_epochs=1`; all other hyperparameters (learning rate multiplier, batch size) used OpenAI's defaults. Gemini 2.5 Pro was fine-tuned via the Vertex AI Supervised Fine-Tuning API with `epochs=1`; all other hyperparameters used Vertex AI's defaults. Prior to training, examples

were filtered to remove any instance whose source text exceeded 1,024 tokens or whose full formatted message exceeded 2,048 tokens (approximately $4\times$ character count), ensuring all training examples fell within the per-turn token budget.

C DSPy Task Signature

The DSPy task signature below defines the input and output fields for the Publishability classification task, including field-level descriptions. It was used as the basis for all three optimizer runs (MIPROv2, GEPA, SIMBA) on Gemini 2.5 Pro, and the resulting SIMBA-optimized prompt was applied to all juror models without per-model re-optimization.

```
class PublishabilityLabeling(dspy.Signature):
    """Evaluate whether
a machine translation is
publishable or
    unpublishable in its current
state.  ..."""

    # --- Input fields ---
    source_locale:  str =
dspy.InputField(
        desc="Language code
for the source language (e.g.
'en-US' for English)"
    )
    target_locale:  str =
dspy.InputField(
        desc="Language code
for the target language (e.g.
'es-ES' for Spanish)"
    )
    source:  str =
dspy.InputField(
        desc="Original text in
the source language"
    )
    target:  str =
dspy.InputField(
        desc="Machine
translation of the source text
in the target language"
    )
    glossary:  list[dict[str,
str]] = dspy.InputField(
        desc="Glossary of
required source to target term
mappings.  May be empty."
    )
    publishability_rules:
list[str] = dspy.InputField(
        desc="List of rules that
determine if a translation is
publishable or unpublishable.
May be empty."
    )
    examples:  list[dict[str,
str]] = dspy.InputField(
        desc="Examples of
similar translated strings along
with their publishability label.
May be empty."
    )

    # --- Output fields ---
    reasoning:  str =
dspy.OutputField(
        desc=(
            "Step-by-step
reasoning for the publishability
decision.  Check:  (1) accuracy
of meaning transfer, (2)
glossary compliance, (3)
publishability rule compliance,
(4) fluency and naturalness in
the target language."
        )
    )
    publish_label:
Literal["publishable",
"unpublishable"] =
dspy.OutputField(
        desc="Final label:
'publishable' if the translation
is ready to use as-is,
'unpublishable' if it requires
editing."
    )
```