Two Step Automatic Post Editing of Patent Machine Translation based on Pre-trained Encoder Models and LLMs

Kosei Buma[1] **Takehito Utsuro**[1] **Masaaki Nagata**[2]
[1]University of Tsukuba [2]NTT, Inc.
s2520812_@_u.tsukuba.ac.jp utsuro_@_iit.tsukuba.ac.jp
masaaki.nagata_@_ntt.com

Abstract

We study automatic post-editing for patent translation, where accuracy and traceability are critical, and propose a two-step pipeline that combines a multilingual encoder for token-level error detection with an LLM for targeted correction. As no word-level annotations exist for Japanese–English patents, we create supervised data by injecting synthetic errors into parallel patent sentences and fine-tune mBERT, XLM-RoBERTa, and mDeBERTa as detectors. In the second stage, GPT-4o is prompted to revise translations either freely or under a restricted policy that allows edits only on detector-marked spans. For error detection, evaluation on synthetic errors shows that encoder-based detectors outperform LLMs in both F1 and MCC. For error correction, tests on synthetic, repetition, and omission datasets demonstrate statistically significant BLEU gains over LLM methods for synthetic and repetition errors, while omission errors remain challenging. Overall, pairing compact encoders with an LLM enables more accurate and controllable post-editing for key patent error types, reducing unnecessary rewrites via restricted edits. Future work will focus on strengthening omission modeling to better detect and correct missing content.

1 Introduction

Recent advances in large language models (LLMs) have enabled powerful multi-step reasoning approaches, such as LLMRefine (Xu et al., 2024), which iteratively refine translation outputs through repeated analysis and correction. More ambitious designs, like Google's recent multi-stage pipeline (Briakou et al., 2024), extend this paradigm even further. However, not all components of a machine translation pipeline need to rely exclusively on LLMs. In particular, error detection can often be performed more accurately and with far lower computational cost using pre-trained transformer encoders (Obeidat et al., 2025). Lukito et al. (2024) demonstrate that, in a classification task detecting connective language—defined as language that facilitates engagement, understanding, and conversation—across social media platforms, a BERT-based classifier significantly outperforms GPT-3.5 Turbo in precision, recall, and F1-score.

In this paper, we present a two-stage translation refinement method (Figure 1) that combines token-level error detection with LLM-based correction. In the first stage, we fine-tune a pre-trained multilingual transformer encoder to identify token-level errors. Because no error-annotated dataset exists for Japanese–English patent translation, we construct a synthetic training set by injecting artificial errors into target-side sentences of parallel patent data. This enables the encoder to learn how to detect mistranslations at the token level. In the second stage, an LLM (GPT-4o[1]) (OpenAI et al., 2024) corrects the translations based on the detected error tags.

We evaluate our method in the patent domain, where translation accuracy has particularly high stakes due to legal and technical requirements, making post-editing especially important. For error detection, we evaluated the fine-tuned multilingual transformer encoder on Japanese–English and English–Japanese patent datasets. The model achieved higher F1 and Matthews correlation coefficient (MCC) scores than an LLM-based approach, demonstrating its superior capability in identifying mistranslations at the token level. For translation correction, experiments on three dataset types—artificially corrupted sentences, repetitive-error sets, and omission scenarios—show that our hybrid strategy, using a compact transformer encoder for detection followed by LLM-based targeted correction, outperforms purely LLM-based

[1]All GPT-4o results are obtained using the `gpt-4o-2024-08-06` model version.

Proceedings of the IJCNLP-AACL 2025 Student Research Workshop, pages 218–231

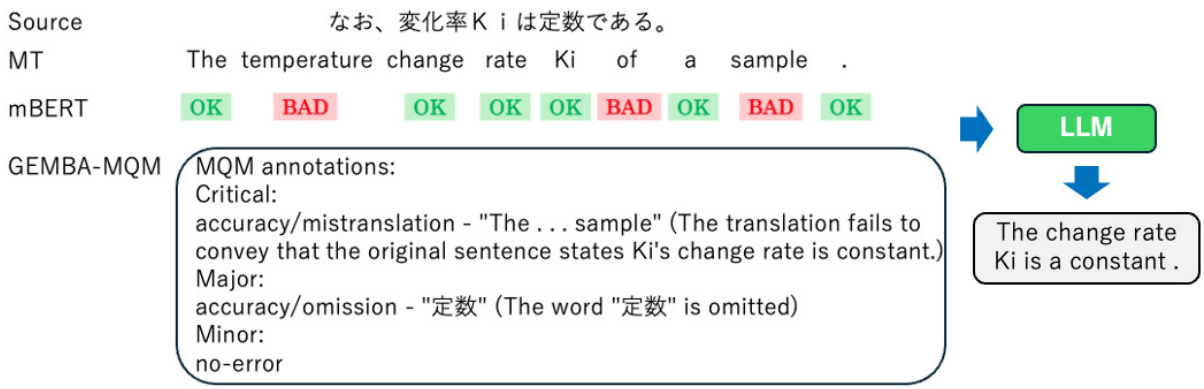

Figure 1: Overview of the Proposed Method.The first stage performs mistranslation detection, and the second stage conducts mistranslation correction.

approaches in BLEU (Papineni et al., 2002) scores. However, omission errors remain difficult to detect and correct, indicating that future work should explore more effective integration of LLM reasoning with dedicated detection modules. We conclude that while multi-step LLM reasoning is powerful, selectively integrating compact transformer encoders can yield more accurate and efficient solutions for machine translation error detection and correction. In summary, our contributions are threefold:

- By fine-tuning encoder models, we achieved higher accuracy than a state-of-the-art LLM on the error detection task, despite LLMs generally showing strong performance across tasks.
- By creating synthetic error-injected patent sentence data, we enabled supervised training of an error detection model without the need for manually annotated datasets.
- Our proposed encoder–LLM hybrid method achieved statistically significant improvements in translation quality compared to LLM-only baselines.

2 Related Work

2.1 Word-Level Quality Estimation in Translation

Word-level QE is commonly formulated as tagging each MT token (and gap positions) with OK/BAD labels, a setup consolidated through the WMT shared tasks and their findings reports over multiple years (Specia et al., 2018; Fonseca et al., 2019; Zerva et al., 2022). This formulation has catalyzed neural approaches and standardized evaluation at the token/gap level without reference translations.

Among early neural architectures, the Predictor–Estimator framework explicitly separates a word predictor trained on large parallel data from a QE estimator trained on annotated QE data, achieving top performance at WMT17 (Kim et al., 2017). Its design influenced subsequent open-source toolkits such as OpenKiwi, which implements state-of-the-art QE systems for word- and sentence-level tasks in a unified PyTorch framework (Kepler et al., 2019). Building on cross-lingual pretrained encoders, Ranasinghe et al. (2020) proposed TransQuest, which attained state-of-the-art results in WMT20 and demonstrated strong cross-lingual transfer.

Closer to our setting, Wei et al. (2022) propose a supervised word-level QE model based on `bert-base-multilingual-cased` (mBERT): given the concatenation of source and MT, a regression head estimates the probability that each MT token is tagged as BAD. We adopt this supervised, token-level formulation for the patent domain, where terminology and style diverge from general-domain WMT data. Beyond a single language pair, multilingual transformer encoders have also shown promising cross-lingual generalization for word-level QE (Ranasinghe et al., 2021).

In parallel, learned MT metrics have moved from sentence-level scores toward span-level feedback. Rei et al. (2022) introduce COMET, while Guerreiro et al. (2024) extend it to xCOMET, which provides sentence-level evaluation and error-span attribution with strong WMT performance. For robustness analysis, Alves et al. (2022) propose SMAUG, a synthetic error generator introducing controlled perturbations (e.g., hallucinations, deletions, mistranslations) to stress-test metrics. Unlike

xCOMET, which uses synthetic errors primarily for metric robustness, we leverage synthetic errors as supervision to train a token-level detector that subsequently guides LLM-based correction.

2.2 LLM-based Quality Evaluation

Large language models (LLMs) have recently been adopted as reference-free, span-level evaluators for machine translation (MT). Kocmi and Federmann (2023) introduce GEMBA-MQM, a GPT-4–based evaluation method that uses a fixed 3-shot prompt to identify error spans and types following the MQM framework (Lommel et al., 2013), without requiring reference translations; their results show strong correlations with human MQM judgments at system and segment levels in WMT23 settings.

At the same time, recent meta-evaluations highlight limitations of LLM-based evaluators. LLM-based metrics show limited robustness; this raises concerns about bias and stability. Broader analyses caution that LLM judges can be sensitive to prompt choices and sometimes conflate evaluation criteria, affecting reliability (Bavaresco et al., 2025). These findings motivate using LLM-based evaluation with care and, when possible, complementing it with interpretable span-level feedback or learned metrics.

In our study we employ LLMs primarily as detectors/correctors rather than as final evaluators: we use GEMBA-MQM–style prompting as one of the error detectors and then perform post-editing with an LLM, while reporting standard automatic metrics (e.g., BLEU) for quantitative evaluation. This design choice balances the interpretability and flexibility of LLMs with established, reproducible evaluation protocols.

2.3 Post-Editing in Machine Translation

Deguchi et al. (2024) propose a Detector–Corrector framework that decomposes Automatic Post-Editing (APE) into two interpretable stages: an XLM-RoBERTa detector performing three binary tagging tasks—MT-tag, MT-gap, and SRC-tag—to localize error spans, followed by a corrector which edits only the detected spans. Their edit-based pipeline improves TER and enhances explainability by tying edits to explicit detector rationales. Our work adopts the same two-stage intuition but replaces the detector with multilingual transformer encoders fine-tuned on patent-domain supervision and couples them with an LLM corrector instructed to modify only detector-marked spans.

In parallel, LLM-based post-editing has emerged. Ki and Carpuat (2024) guide an LLM with external MQM-style feedback—at varying granularities from generic scores to fine-grained span/type annotations—and show consistent improvements in TER, BLEU, and COMET on Zh–En, En–De, and En–Ru, with fine-grained feedback yielding the strongest gains. Orthogonally, Xu et al. (2024) introduce LLMRefine, which iteratively pinpoints defects with a learned feedback model and refines hypotheses, improving translation quality.

3 Mistranslation Detection

3.1 Mistranslation Detection Using Encoders

In this study, we utilize mBERT[2], XLM-RoBERTa[3], mDeBERTa[4] (He et al., 2023), which are pre-trained multilingual transformer encoders, to perform token-level quality estimation in machine translation. Specifically, we leverage the pre-trained knowledge of encoder models to detect translation errors and assign appropriate error labels to each token.

For training encoder models, we follow the data augmentation method proposed by Deguchi et al. (2024) and generate synthetic error data by sampling 10,000 sentences from the NTCIR-7 (Fujii et al., 2008) (1,798,571 sentence pairs) and NTCIR-8 (Fujii et al., 2010) (3,186,284 sentence pairs) patent parallel corpora. We sample 10,000 sentence pairs and generate synthetic errors for both translation directions. The same 10,000 pairs are split into 8,000 for training, 1,000 for development, and 1,000 for testing, before applying the following operations:

- Deletion: Delete tokens with a probability of 5%
- Insertion: Insert tokens with a probability of 10%
- Replacement: Replace tokens with a probability of 30%

The probabilities of these operations are determined in accordance with Deguchi et al. (2024). For insertion and replacement, we adopt a mask-filling approach using mBERT. We insert [MASK]

[2] https://huggingface.co/google-bert/bert-base-multilingual-cased

[3] https://huggingface.co/FacebookAI/xlm-roberta-base

[4] https://huggingface.co/microsoft/mdeberta-v3-base

tokens at the target positions and let mBERT generate candidate substitutions using the fill-mask prediction head. From the top-k predictions returned by the model (we set $k = 5$), we intentionally choose the token with the *lowest* predicted probability so as to maximize the divergence from the original token. This token is then inserted or substituted to produce an artificial error. After generating the corrupted sentence, we annotate the manipulated tokens with the BAD tag and all other tokens with the OK tag to construct supervised training dataset.

Using this method, we generate 8,000 annotated sentences for training encoder models. Training uses the Hugging Face Trainer[5] with `num_train_epochs = 10` and `per_device_train_batch_size = 2`. Unless otherwise specified, we keep the Hugging Face defaults for optimizer and scheduler (AdamW, learning rate = 5×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, linear scheduler).

To assess the effectiveness of the constructed training data, we conducted additional experiments with mBERT under different dataset conditions. Details are provided in Appendix A.

3.2 Mistranslation Detection Using LLM

We adopt GEMBA-MQM, a GPT-based evaluation method proposed by Kocmi and Federmann (2023), for mistranslation detection using large language models (LLMs). Based on the GEMBA-MQM framework, we perform error detection under the following two settings:

- 0-shot: Error detection is performed without any prior examples.
- 3-shot: Error detection is performed using three language-independent examples, following exactly the same examples provided by Kocmi and Federmann (2023).

Among these (Kocmi and Federmann, 2023), the 3-shot setting has been reported to achieve the highest error detection accuracy using GPT-4.

The mistranslation detection using encoder models and LLM serves as a preprocessing step for the subsequent translation correction. By utilizing the detection results, we aim to enhance the accuracy of the translation correction process.

[5] https://huggingface.co/docs/transformers/main_classes/trainer

4 Mistranslation Correction Using LLM

By providing both the source sentence and its translation as input, the LLM analyzes translation errors and generates appropriate corrections. Specifically, the LLM closely analyzes the detected erroneous parts, explains the nature of the errors and their locations, and generates corrected translations based on this analysis. By explicitly stating the reasoning behind each correction, the LLM enhances the transparency of the correction process and makes the translation refinement more interpretable.

Furthermore, in this study, we propose a method that utilizes the mistranslation detection results obtained from the encoder described in the previous section as input for translation correction using an LLM. In this experiment, we tested two types of prompts: one instructing the LLM to perform translation correction with reference to the first-stage error detection results, and another instructing it to correct only the segments identified as erroneous in the first stage, leaving all other parts unchanged. By incorporating the detection outputs from either the LLM or encoder model, we aim to further improve the accuracy of translation correction.

The prompt for the proposed method is provided in Appendix B.

5 Evaluation

5.1 Dataset

In this study, we focus on mistranslations, repetitions, and omissions. These error types are not only frequently observed in patent translations but also critically impact semantic fidelity, which is of utmost importance in the context of patent documents. we evaluate our proposed method using the following three types of datasets:

- Mistranslation patent dataset (Synthetic)
- Repetition error patent claim dataset (Non-Synthetic)
- Omission error patent claim dataset (Non-Synthetic)

The synthetic error patent data is generated by introducing artificial errors into Japanese-English parallel patent sentences from NTCIR-7 and NTCIR-8 using the method described in Section 3.1. We evaluate the detection and correction capabilities of our method using 200 sentences from the synthetic error patent dataset.

Model	Label	Precision	Recall	F1
GPT-4o (0-shot)	OK	0.843	0.077	0.142
	BAD	0.389	0.976	0.556
	TOTAL	F1: 0.298,	MCC: 0.111	
GPT-4o (3-shot)	OK	0.755	0.268	0.395
	BAD	0.409	0.853	0.553
	TOTAL	F1: 0.454,	MCC: 0.141	
mBERT	OK	0.855	0.883	0.869
	BAD	0.785	0.740	0.762
	TOTAL	F1: 0.830*,	MCC: 0.631*	
XLM-RoBERTa	OK	0.924	0.924	0.924
	BAD	0.868	0.868	0.868
	TOTAL	F1: 0.903*,	MCC: 0.792*	
mDeBERTa	OK	0.935	0.944	**0.940**
	BAD	0.901	0.887	**0.894**
	TOTAL	F1: **0.923***,	MCC: **0.834***	

(a) Japanese-English Translation

Model	Label	Precision	Recall	F1
GPT-4o (0-shot)	OK	0.804	0.287	0.423
	BAD	0.415	0.879	0.563
	TOTAL	F1: 0.474,	MCC: 0.190	
GPT-4o (3-shot)	OK	0.709	0.503	0.588
	BAD	0.419	0.634	0.504
	TOTAL	F1: 0.558,	MCC: 0.132	
mBERT	OK	0.870	0.880	0.875
	BAD	0.784	0.769	0.776
	TOTAL	F1: 0.839*,	MCC: 0.655*	
XLM-RoBERTa	OK	0.918	0.935	0.926
	BAD	0.881	0.852	0.866
	TOTAL	F1: 0.904*,	MCC: 0.792*	
mDeBERTa	OK	0.936	0.946	**0.941**
	BAD	0.903	0.885	**0.894**
	TOTAL	F1: **0.924***,	MCC: **0.835***	

(b) English-Japanese Translation

Table 1: Mistranslation Detection Evaluation on Synthetic Errors. $*$ indicates a statistically significant difference from the GPT-4o(3-shot) ($p < 0.05$).

In addition, to assess correction accuracy for repetition and omission errors, we extract sentences from Japanese–English translations of patent claims generated by a Transformer (Vaswani et al., 2023) based on the following criteria:

- Repetition error sentences: Translated sentences that are more than twice as long as the reference translation
- Omission error sentences: Translated sentences that are less than half the length of the reference translation

We then evaluate the correction performance using patent claims extracted from patent documents published in 2021. To ensure the quality of the parallel data, we compute sentence similarity between the source and reference translations using LaBSE embeddings (Feng et al., 2022), and extract only those pairs with similarity scores between 0.8 and 0.98. As a result, we use 200 repetition error sentences and omission error sentences for evaluation.

Further details of the datasets, including the number of sentences, tokens, and other statistics, are provided in Appendix C.

5.2 Evaluation Procedure

5.2.1 Mistranslation Detection

Each token in the translated sentence is labeled with a BAD tag at erroneous positions, allowing for token-level evaluation. Both encoder models and the LLM perform tagging in the same manner as illustrated on the left side of Figure 1. For Japanese tokenization, we employed MeCab[6] together with the UniDic dictionary[7].

We evaluate translation error detection using the following models:

1. LLM(GPT-4o) - 0-shot
2. LLM(GPT-4o) - 3-shot
3. mBERT
4. XLM-RoBERTa
5. mDeBERTa

All experiments involving GPT-4o—both in detection and correction—use greedy decoding (temperature = 0.0), with all other parameters kept at their provider defaults.

We report F1 score and Matthews Correlation Coefficient (MCC) as our evaluation metrics.

5.2.2 Mistranslation Correction

For the evaluation of error correction, we use three types of data: patent sentences with artificially introduced errors, patent claim sentences of repetition errors and patent claim sentences of omission errors. Error correction is performed using an LLM, where the input consists of the source sentence, the translated sentence, and the error detection results from encoder model or LLM.

The combinations of models used in the evaluation are as follows:

1. **No Correction**: The raw MT output is evaluated without any post-editing.

[6] https://taku910.github.io/mecab/
[7] https://clrd.ninjal.ac.jp/unidic/

	Method	Synthetic (Ja→En)		Synthetic (En→Ja)		Repetition		Omission	
		BLEU	Δ	BLEU	Δ	BLEU	Δ	BLEU	Δ
1	No Correction	31.69	−14.72	32.63	−6.46	21.49	−4.79	19.09	−45.40
2	LLM-only Correction	47.25	+0.84	**40.35**	+1.26*	26.70	+0.42	62.50	−1.99
3	LLM Detection (GEMBA-MQM, 0-shot) + LLM Correction	43.88	−2.53	37.58	−1.51	26.16	−0.12	62.31	−2.18
4	**Baseline**: LLM Detection (GEMBA-MQM, 3-shot) + LLM Correction	46.41	–	39.09	–	26.28	–	**64.49**	–
5	**Proposed**: mBERT Detection + LLM Correction	48.44	**+2.03***	39.10	+0.01	27.42	**+1.14***	59.73	−4.76
6	**Proposed**: XLM-RoBERTa Detection + LLM Correction	**48.83**	**+2.42***	39.21	+0.12	27.41	**+1.13***	56.38	−8.11
7	**Proposed**: mDeBERTa Detection + LLM Correction	48.03	**+1.62***	39.22	+0.13	**28.15**	**+1.87***	57.73	−6.76

(a) Unrestricted post-editing: the LLM may modify any part of the MT output. Δ is computed as the difference from the baseline's score (line 4).

	Method	Synthetic (Ja→En)		Synthetic (En→Ja)		Repetition		Omission	
		BLEU	Δ	BLEU	Δ	BLEU	Δ	BLEU	Δ
1	No Correction	31.69	−14.72	32.63	−6.46	21.49	−4.79	19.09	−45.40
2	LLM-only Correction	47.25	+0.84	40.35	+1.26*	26.70	+0.42	62.50	−1.99
3	LLM Detection (GEMBA-MQM, 0-shot) + LLM Correction	44.05	−2.36	39.96	+0.87	**27.25**	+0.97	62.82	−1.67
4	LLM Detection (GEMBA-MQM, 3-shot) + LLM Correction	48.14	+1.73*	42.05	+2.96*	26.58	+0.30	**65.62**	+1.13*
5	**Proposed**: mBERT Detection + LLM Correction	49.76	**+3.35***	42.61	**+3.52***	23.46	−2.82	28.00	−36.49
6	**Proposed**: XLM-RoBERTa Detection + LLM Correction	**50.71**	**+4.30***	**43.76**	**+4.67***	23.15	−3.13	25.57	−38.92
7	**Proposed**: mDeBERTa Detection + LLM Correction	50.69	**+4.28***	**43.76**	**+4.67***	22.63	−3.65	22.60	−41.89

(b) Restricted post-editing: the LLM is allowed to modify only spans detected as erroneous. Δ is computed as the difference from the baseline's score (line 4 of Table 2a).

Table 2: Translation correction BLEU scores under unrestricted and restricted post-editing settings. $*$ on Δ indicates a statistically significant difference from the baseline ($p < 0.05$).

2. **LLM-only Correction**: Translation correction in a single step using only LLM (GPT-4o), without prior error detection.
3. **LLM Detection (GEMBA-MQM, 0-shot) + LLM Correction**: Error detection with LLM (GPT-4o) using GEMBA-MQM (0-shot), followed by translation correction with LLM (GPT-4o).
4. **LLM Detection (GEMBA-MQM, 3-shot) + LLM Correction**: Error detection with LLM (GPT-4o) using GEMBA-MQM (3-shot), followed by translation correction with LLM (GPT-4o).
5. **mBERT Detection + LLM Correction**: Error detection with mBERT (token-level tagging), followed by translation correction with LLM (GPT-4o).
6. **XLM-RoBERTa Detection + LLM Correction**: Error detection with XLM-RoBERTa (token-level tagging), followed by translation correction with LLM (GPT-4o).
7. **mDeBERTa Detection + LLM Correction**: Error detection with mDeBERTa (token-level tagging), followed by translation correction with LLM (GPT-4o).

Baseline *LLM Detection (GEMBA-MQM, 3-shot) + LLM Correction (unrestricted).* The LLM performs error detection with GEMBA-MQM (3-shot), and the subsequent correction step allows edits to *any* part of the translation (unrestricted).

Proposed *Encoder-based Detection + LLM Correction (restricted).* Error detection is performed by an encoder model (mBERT, XLM-RoBERTa, or mDeBERTa), and the correction step is *restricted* to only the spans flagged as erroneous by the detector; all other tokens must remain unchanged.

For Japanese target sentences in English-Japanese translation correction, the corrected outputs sometimes contained tokenized text with

Source Sentence:
ステップＳ１１において、プライマリプーリ１１への入力トルクを計算する。
Reference Translation:
In a step S11 , an input torque to the primary pulley 11 is calculated .
Synthetic Error Sentence:
In a processing stepd , The input torque to be primary pulley 11 is achieved :
Proposed Method:
In step S11, the input torque to primary pulley 11 is calculated.

Table 3: Correction Examples of Synthetic Errors by the Proposed Method

spaces between characters, so we removed these spaces. A comparison of results before and after space removal is provided in Appendix D.

The corrected translations are evaluated using BLEU scores computed with sacreBLEU (v2.4.3) (Post, 2018). BLEU measures the n-gram overlap between a system translation and reference translations, and is widely used as an automatic metric for translation quality. Since BLEU is often prioritized in domains requiring highly faithful translations, such as patents, we adopt this metric for our evaluation. To assess whether the BLEU score improvements reported in Table 2 are statistically significant, we used the paired-bootstrap resampling test implemented in SacreBLEU (via the –paired-bs option).

5.3 Evaluation Results

5.3.1 Mistranslation Detection Evaluation

On the synthetic-error evaluation (Table 1), fine-tuned encoder models significantly outperform LLM-based detection in both directions. mDeBERTa yields the best performance (Ja→En: F1 = 0.923, MCC = 0.834; En→Ja: F1 = 0.924, MCC = 0.835), followed by XLM-RoBERTa and mBERT. In contrast, using GPT-4o as a detector —even with 3-shot prompting (Ja→En: F1 = 0.454, MCC = 0.141; En→Ja: F1 = 0.558, MCC = 0.132; 0-shot is lower). An analysis of GPT-4o's output revealed that it tended to assign the BAD tag to most tokens. As a result, while the recall for BAD tags was relatively high, the recall for OK tags dropped significantly.

These results confirm that supervised fine-tuning of compact encoders using synthetically generated error data is more effective for token-level mistranslation detection than prompting an LLM. As human-annotated data in the patent domain is not publicly available, we further report experiments on the WMT21 QE dataset (Specia et al., 2021) in the En→Ja direction, and the results are provided in Appendix E.

5.3.2 Mistranslation Correction Evaluation

As shown in Table 2, the best-performing approach depends on the error type and language direction. Detector–corrector pipelines consistently improve over the No Correction baseline, while our encoder models-based detector with an LLM corrector is competitive but not uniformly superior to all alternatives.

For synthetic errors (Ja→En), our proposed methods outperform the LLM-only corrector and LLM-based detector methods. The strongest result is obtained with XLM-RoBERTa detection + LLM correction (48.83 BLEU / 50.71), with our mBERT detection + LLM correction close behind (48.44 / 49.76), both surpassing the LLM-only corrector (47.25 / 47.25). The qualitative example in Table 3 show that these pipelines reliably fix mistranslations in the manipulated inputs, indicating that token-level error tags are effective cues for the LLM corrector.

For synthetic errors (En→Ja), when the prompt instructs the LLM to revise the translation with reference to the first-stage error detection results, the LLM-only correction achieves the highest BLEU (40.35). However, when the prompt is modified to instruct the LLM to correct only the spans identified in the first-stage detection (leaving other parts unchanged), our proposed method surpasses the LLM-only methods, achieving the highest BLEU (43.76 with XLM-RoBERTa or mDeBERTa detection). This trend is also observed for synthetic errors (Ja→En), where the second prompt formulation yields higher scores than the first. These results suggest that the high accuracy of the first-stage detection contributes positively to the overall translation correction quality.

For repetition errors, the highest BLEU is achieved by mDeBERTa detection + LLM correction (28.15), followed by our mBERT detection +

Source Sentence:
a.配電ハードウェアの構成部品として、少なくとも1つの受動電磁センサをインストールするステップと、 b.
Reference Translation:
a.Installing at least one Passive Electromagnetic Sensor as a component of distribution hardware;b.
Machine Translation Sentence:
a. installing at least one passive electromagnetic sensor as a component part of the electrical distribution hardware;b. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;c. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;d. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;e. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;f. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;g. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;g. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;g. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;g. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;g. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;g. controlling the at least one passive electromagnetic sensor to emit electromagnetic radiation;g
Proposed Method:
a. installing at least one passive electromagnetic sensor as a component part of the electrical distribution hardware; b.

Table 4: Correction Examples of Repetition Errors by the Proposed Method

Source Sentence:
前記NK細胞は、血液または細胞株に由来し、好ましくは、細胞株に由来し、より好ましくは、前記細胞株に由来するNKはNK92細胞株であることを特徴とする 請求項12に記載の免疫細胞。
Reference Translation:
The immune cell of claim 12, wherein the NK cell is derived from blood or a cell line; preferably, from a cell line; and more preferably, the NK cell from a cell line is NK92 cell line.
Machine Translation Sentence:
The immune cell of claim 12, wherein the NK cell is derived from blood or a cell line.
Proposed Method:
The immune cell of claim 12, characterized in that the NK cell is derived from blood or a cell line, preferably from a cell line, and more preferably from the aforementioned cell line, specifically the NK92 cell line.

Table 5: Correction Examples of Omission Errors by the Proposed Method

LLM correction (27.42). The example in Table 4 confirm that proposed method remove duplicated spans while preserving punctuation and other formatting. These findings suggest that the proposed method is also effective in handling repetition errors. While unrestricted edits generally yield higher BLEU scores, restricting edits to detector-identified spans leads to a performance drop across most encoder–LLM configurations. This suggests that it is important to balance the use of detection outputs with the LLM's inherent correction capability. Based on the detection results, this decline is likely due to the first-stage detection failing to identify repetition errors with sufficient accuracy, leading to a drop in performance when relying too heavily on these detection outputs. Therefore, improving both precision and recall in error detection is essential for the restricted-edit pipeline to match or exceed the performance of the unrestricted approach. Patent claims often contain longer sentences than general patent text, so it may be necessary to train the encoder model on longer sentences to better handle lengthy sentences such as repetitions.

In contrast, for omission errors, the strongest performance comes from LLM detection (3-shot) + LLM correction (64.49 / 65.62), with the LLM-

only corrector next. As shown in Table 2a, the BLEU score of the uncorrected translations was 19.09, whereas the proposed method achieved a significantly higher score of 59.73. Table 5 presents the example of omission error corrections, illustrating cases where the proposed method successfully recovers missing content in patent translations. However, our encoder-based detector lags on this error type. Unlike mistranslations or repetitions—which are anchored to existing target tokens—omissions are not directly observable on the target side via token tags. This likely limits target-side tagging, whereas sequence-/alignment-aware detection (e.g., identifying source tokens without target alignments) is better suited to omissions. Incorporating alignment-based signals is therefore a promising direction to broaden omission coverage in future work.

6 Conclusion

This study demonstrated that the combination of pre-trained multilingual transformer encoder model, trained on patent texts for mistranslation detection, and LLM-based correction led to statistically significant improvements in BLEU scores, outperforming other methods in handling mistranslations and repetition errors. In particular, the high-precision error detection achieved by encoder models supported the LLM in correcting erroneous tokens, contributing to overall improvements in translation quality.

Moreover, by training on synthetically generated patent data, we showed that it is possible to train an error detection model without relying on human-annotated data. These findings suggest that an encoder-based model, when trained with high-quality data, can outperform LLMs—which typically excel in a wide range of tasks—in specific scenarios such as error detection in patent translation.

On the other hand, for omission errors, the model that performed both detection and correction solely with an LLM outperformed the proposed method, highlighting a limitation in the current use of token-level tagging. These results indicate that optimizing correction strategies and error representation methods based on the type of error is essential for further improving translation quality.

Limitations

While our proposed two-step method achieves promising results in detecting and correcting translation errors in patent documents, several limitations remain.

First, our study focuses on mistranslations, repetitions, and omissions. While these types are critical in patent translation, other important categories—such as terminology misuse and grammatical inconsistencies—remain unexamined. Prior work has shown that comprehensive MT evaluation requires explicit error analysis across diverse categories, as formalized in the MQM framework (Freitag et al., 2021). Motivated by this, future work will investigate improved methods for constructing synthetic data that more faithfully capture a broader range of error types.

Second, all experiments were conducted in the Japanese–English patent domain. Thus, the generalizability of our approach to other domains or language pairs remains unverified. We plan to apply our method to diverse translation settings to evaluate its robustness.

Third, our evaluation used relatively small datasets, due to the limited availability of high-quality, domain-specific parallel data. Larger-scale validation would help confirm the effectiveness of our approach.

Finally, token-level tagging was less effective for omissions, likely due to their broader contextual nature. To improve this, we will explore incorporating alignment-based signals and increase training data diversity to better capture omission patterns.

References

Duarte Alves, Ricardo Rei, Ana C Farinha, José G. C. de Souza, and André F. T. Martins. 2022. Robust MT evaluation with sentence-level multilingual augmentation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 469–478, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, Andre Martins, Philipp Mondorf, Vera Neplenbroek, Sandro Pezzelle, Barbara Plank, David Schlangen, Alessandro Suglia, Aditya K Surikuchi, Ece Takmaz, and Alberto Testoni. 2025. LLMs instead of human judges? a large scale empirical study across 20 NLP evaluation tasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*

(Volume 2: Short Papers), pages 238–255, Vienna, Austria. Association for Computational Linguistics.

Eleftheria Briakou, Jiaming Luo, Colin Cherry, and Markus Freitag. 2024. Translating step-by-step: Decomposing the translation process for improved translation quality of long-form texts. In *Proceedings of the Ninth Conference on Machine Translation*, pages 1301–1317, Miami, Florida, USA. Association for Computational Linguistics.

Hiroyuki Deguchi, Masaaki Nagata, and Taro Watanabe. 2024. Detector–corrector: Edit-based automatic post editing for human post editing. In *Proceedings of the 25th Annual Conference of the European Association for Machine Translation (Volume 1)*, pages 191–206, Sheffield, UK. European Association for Machine Translation (EAMT).

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Erick Fonseca, Lisa Yankovskaya, André F. T. Martins, Mark Fishel, and Christian Federmann. 2019. Findings of the WMT 2019 shared tasks on quality estimation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

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.

Atsushi Fujii, Masao Utiyama, Mikio Yamamoto, and Takehito Utsuro. 2008. Overview of the patent translation task at the NTCIR-7 workshop. In *Proc. 7th NTCIR*, pages 389–400.

Atsushi Fujii, Masao Utiyama, Mikio Yamamoto, Takehito Utsuro, Terumasa. Ehara, Hiroshi. Echizen-ya, and Sayori. Shimohata. 2010. Overview of the patent translation task at the NTCIR-8 workshop. In *Proc. 8th NTCIR*, pages 371–376.

Nuno M. Guerreiro, Ricardo Rei, Daan van Stigt, Luisa 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.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *Preprint*, arXiv:2111.09543.

Fabio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, and André F. T. Martins. 2019. OpenKiwi: An open source framework for quality estimation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 117–122, Florence, Italy. Association for Computational Linguistics.

Dayeon Ki and Marine Carpuat. 2024. Guiding large language models to post-edit machine translation with error annotations. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4253–4273, Mexico City, Mexico. Association for Computational Linguistics.

Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. 2017. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In *Proceedings of the Second Conference on Machine Translation*, pages 562–568, Copenhagen, Denmark. Association for Computational Linguistics.

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, Singapore. Association for Computational Linguistics.

Arle Richard Lommel, Aljoscha Burchardt, and Hans Uszkoreit. 2013. Multidimensional quality metrics: a flexible system for assessing translation quality. In *Proceedings of Translating and the Computer 35*, London, UK. Aslib.

Josephine Lukito, Bin Chen, Gina M. Masullo, and Natalie Jomini Stroud. 2024. Comparing a BERT classifier and a GPT classifier for detecting connective language across multiple social media. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 19140–19153, Miami, Florida, USA. Association for Computational Linguistics.

Motasem S Obeidat, Md Sultan Al Nahian, and Ramakanth Kavuluru. 2025. Do llms surpass encoders for biomedical ner? *Preprint*, arXiv:2504.00664.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

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*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on*

Machine Translation: Research Papers, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2020. TransQuest: Translation quality estimation with cross-lingual transformers. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5070–5081, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2021. An exploratory analysis of multilingual word-level quality estimation with cross-lingual transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 434–440, Online. Association for Computational Linguistics.

Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Chrysoula Zerva, Zhenhao Li, Vishrav Chaudhary, and André F. T. Martins. 2021. Findings of the WMT 2021 shared task on quality estimation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 684–725, Online. Association for Computational Linguistics.

Lucia Specia, Frédéric Blain, Varvara Logacheva, Ramón F. Astudillo, and André F. T. Martins. 2018. Findings of the WMT 2018 shared task on quality estimation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 689–709, Belgium, Brussels. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Yizhen Wei, Takehito Utsuro, and Masaaki Nagata. 2022. Extending word-level quality estimation for post-editing assistance. *Preprint*, arXiv:2209.11378.

Wenda Xu, Daniel Deutsch, Mara Finkelstein, Juraj Juraska, Biao Zhang, Zhongtao Liu, William Yang Wang, Lei Li, and Markus Freitag. 2024. LLMRefine: Pinpointing and refining large language models via fine-grained actionable feedback. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1429–1445, Mexico City, Mexico. Association for Computational Linguistics.

Chrysoula Zerva, Frédéric Blain, Ricardo Rei, Piyawat Lertvittayakumjorn, José G. C. de Souza, Steffen Eger, Diptesh Kanojia, Duarte Alves, Constantin Orăsan, Marina Fomicheva, André F. T. Martins, and Lucia Specia. 2022. Findings of the WMT 2022 shared task on quality estimation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 69–99, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

A Effectiveness of the Constructed Training Data

To evaluate the effectiveness of the constructed training data, we compare the tagging performance of the following models:

1. mBERT without fine-tuning
2. mBERT trained on low-quality synthetic data
3. mBERT trained on high-quality synthetic data (proposed model)

The low-quality training data is generated by inserting or replacing tokens using the most likely candidates predicted by mBERT. As these tokens tend to be highly similar to the original tokens, tagging them as BAD degrades the quality of the training data.

As shown in Table 6, our proposed method achieves the highest error detection accuracy, demonstrating the effectiveness of the constructed training data.

B Prompt Template for Translation Correction

This appendix presents the prompts used in the second stage of our method, where the LLM generates corrected translations (Japanese-English) based on the source sentence, the initial translation, and token-level error tags. In this experiment, we tested two types of prompts: one instructing the LLM to perform translation correction with reference to the first-stage error detection results (shown in Table 9), and another instructing it to correct only the segments identified as erroneous in the first stage, leaving all other parts unchanged (shown in Table 10). By incorporating the detection outputs from the encoder model, we aim to further improve the accuracy of translation correction.

C Dataset Statistics

In this appendix, we provide detailed statistics of the datasets used in our experiments. We report

Model	Label	Precision	Recall	F1
mBERT before fine-tuning	OK	0.673	0.140	0.232
	BAD	0.372	0.882	0.523
	TOTAL	F1: 0.338,	MCC: 0.031	
Low-quality Training Data	OK	0.707	0.913	0.797
	BAD	0.694	0.342	0.459
	TOTAL	F1: 0.673,	MCC: 0.320	
Proposed Model	OK	0.855	0.883	**0.869**
	BAD	0.785	0.740	**0.762**
	TOTAL	F1: **0.830**,	MCC: **0.631**	

(a) Japanese-English Translation

Model	Label	Precision	Recall	F1
mBERT before fine-tuning	OK	0.667	0.002	0.004
	BAD	0.362	0.998	0.531
	TOTAL	F1: 0.195,	MCC: 0.002	
Low-quality Training Data	OK	0.747	0.873	0.805
	BAD	0.681	0.478	0.562
	TOTAL	F1: 0.717,	MCC: 0.388	
Proposed Model	OK	0.870	0.880	**0.875**
	BAD	0.784	0.769	**0.776**
	TOTAL	F1: **0.839**,	MCC: **0.655**	

(b) English-Japanese Translation

Table 6: Tagging Accuracy Evaluation of mBERT

	Ja→En			En→Ja		
Dataset	**Sent.**	**Tokens (Ja)**	**Tokens (En)**	**Sent.**	**Tokens (En)**	**Tokens (Ja)**
Training	8,000	268,247	254,239	8,000	246,885	280,313
Development	1,000	37,062	34,128	1,000	33,160	38,670
Test (Mistranslation)	200	7,016	6,869	200	6,680	7,278
Test (Repetition)	200	26,454	19,805	-	-	-
Test (Omission)	200	67,613	11,545	-	-	-

Table 7: Statistics of the datasets used in this study

the number of sentences and tokens for the training, development, and test sets. For the test data, we further break down the statistics by error type: mistranslation, repetition, and omission.

As shown in Table 7, the training and development sets are derived from synthetic error corpora constructed from Japanese–English patent sentences. The test sets include both synthetic errors (mistranslation) and human-annotated patent claim data (repetition and omission).

D Impact of Space Removal in Japanese Translations

In the Japanese target sentences produced during English–Japanese translation correction, some outputs contained tokenized text with spaces inserted between characters. To ensure accurate BLEU calculation and fair comparison, we removed spaces between characters in Japanese outputs prior to scoring. Table 11 shows the comparison of BLEU scores before and after space removal, demonstrating that removing extraneous spaces can lead to score variations due to changes in tokenization.

Model	Label	Precision	Recall	F1
GPT-4o (0-shot)	OK	0.757	0.957	0.845
	BAD	0.280	0.052	0.087
	TOTAL	F1: 0.660,	MCC: 0.018	
GPT-4o (3-shot)	OK	0.766	0.945	0.846
	BAD	0.390	0.108	0.169
	TOTAL	F1: 0.681,	MCC: 0.091	
mDeBERTa	OK	0.808	0.891	**0.847**
	BAD	0.506	0.346	**0.411**
	TOTAL	F1: **0.741**,	MCC: **0.272**	

Table 8: Mistranslation Detection Evaluation on WMT21 En→Ja QE Dataset

E Results on the WMT21 En→Ja QE Dataset

To further evaluate our method on human-annotated data, we conducted experiments using the WMT21 quality estimation (QE) dataset in the En→Ja direction. From the dataset, 800 sentences were used for training and 100 sentences for evaluation. We compared the performance of the fine-tuned mDeBERTa model with GPT-4o under the same conditions. The results are presented in Table 8.

From the results, we observed that the fine-tuned mDeBERTa achieved the highest detection accuracy. This suggests that, even on human-annotated

System Prompt

You are a translation checker.
You will be given:
1) A Japanese sentence (source text).
2) An English sentence (the current translation).
3) A list of token-level annotations (BAD/OK) for the English sentence.
Your tasks are:
1. Identify translation errors or inaccuracies in the English sentence relative to the Japanese source.
- Use the BAD/OK annotation list as a reference, but also rely on your own judgment.
2. Propose corrections or improvements for each identified error.
3. Provide a final, corrected English translation that reflects all improvements.
Output Format:
`[Translation Errors]`
- (1) <具体的にどの部分が誤りか、どのように修正すべきか>
- (2) <...>
...
`[Corrected Translation]`
<最終的に修正を反映した正しい英文>
Constraints:
- Do not provide explanations or commentary beyond what is requested in the Output Format.
- Keep your output concise and organized.

User Prompt
Japanese source sentence: {source_text.strip()} English translation to check: {translated_text.strip()} Token-level annotation: {annotation_list.strip()} Please: 1. List errors and their corrections under `[Translation Errors]`. 2. Provide the corrected translation under `[Corrected Translation]`.

Table 9: Prompt for translation correction with reference to the encoder-based error detection results, without restrictions on the parts to be corrected

data, encoder-based models can surpass LLMs in error detection accuracy. Compared to the synthetic-error evaluation results in Table 1, the scores are lower for two reasons. First, the amount of human-annotated training data is limited, as only a small portion of such data has been made publicly available. Second, human-annotated data is inherently more challenging than synthetic data. Therefore, constructing synthetic data that more closely approximates human annotations represents an important future direction.

System Prompt

You are a translation checker.
You will be given:
1) A Japanese sentence (source text).
2) An English sentence (the current translation).
3) A list of token-level annotations (BAD/OK) for the English sentence.
Your tasks are:
1. Based only on the BAD/OK annotation list, identify the tokens marked as BAD in the English translation.
2. Propose corrections or improvements only for the BAD tokens. Do not introduce corrections for tokens marked as OK.
3. Provide a final, corrected English translation that reflects only the necessary changes.
Output Format:
`[Translation Errors]`
- (1) <具体的にどの部分が誤りか、どのように修正すべきか>
- (2) <...>
...
`[Corrected Translation]`
<最終的に修正を反映した正しい英文>
Constraints:
- Do not consider or correct any parts of the translation other than the tokens marked as BAD.
- Do not provide explanations or commentary beyond what is requested in the Output Format.
- Keep your output concise and organized.

User Prompt

Japanese source sentence:
{source_text.strip()}
English translation to check:
{translated_text.strip()}
Token-level annotation:
{annotation_list.strip()}
Please:
1. List errors and their corrections under `[Translation Errors]`.
2. Provide the corrected translation under `[Corrected Translation]`.

Table 10: Prompt for correcting only the segments identified as erroneous by the encoder-based error detection, leaving all other parts unchanged

	Method	Before	After
1	No Correction	32.63	32.63
2	LLM-only Correction	40.35	40.35
3	LLM Detection (GEMBA-MQM, 0-shot) + LLM Correction	37.58 / 39.96	37.58 / 39.96
4	LLM Detection (GEMBA-MQM, 3-shot) + LLM Correction	39.09 / 42.05	39.09 / 42.05
5	mBERT Detection + LLM Correction	39.10 / 42.62	39.10 / 42.61
6	XLM-RoBERTa Detection + LLM Correction	39.21 / 43.64	39.21 / 43.76
7	mDeBERTa Detection + LLM Correction	39.21 / 43.70	39.22 / 43.76

Table 11: Effect BLEU of removing extra spaces in Japanese target sentences. Each cell shows x/y, where x is the LLM correction with *unrestricted* edits (may modify any part) and y is the LLM correction *restricted* to correcting only the errors detected. "Before" denotes the raw corrected outputs containing spaces between characters, and "After" denotes the same outputs with these spaces removed.