Structured Document Translation via Format Reinforcement Learning

Haiyue Song[1], **Johannes Eschbach-Dymanus**[2], **Hour Kaing**[1], **Sumire Honda**[2], **Hideki Tanaka**[1], **Bianka Buschbeck**[2], **Masao Utiyama**[1]
[1]National Institute of Information and Communications Technology, Japan [2]SAP, Germany
{haiyue.song,hour_kaing,hideki.tanaka,mutiyama}@nict.go.jp
{johannes.eschbach-dymanus,sumire.honda,bianka.buschbeck}@sap.com

Abstract

Recent works on structured text translation remain limited to the sentence level, as they struggle to effectively handle the complex document-level XML or HTML structures. To address this, we propose **Format Reinforcement Learning** (FORMATRL), which employs Group Relative Policy Optimization on top of a supervised fine-tuning model to directly optimize novel structure-aware rewards: 1) TreeSim, which measures structural similarity between predicted and reference XML trees and 2) Node-chrF, which measures translation quality at the level of XML nodes. Additionally, we apply StrucAUC, a fine-grained metric distinguishing between minor errors and major structural failures. Experiments on the SAP software-documentation benchmark demonstrate improvements across six metrics and an analysis further shows how different reward functions contribute to improvements in both structural and translation quality.[1]

1 Introduction

Translating structured documents such as software manuals is essential for product localization. As shown in Figure 1, they carry markup that defines layout and interactive elements, making structural fidelity as important as content translation quality.

Until the advent of large language models (LLMs), the most prevalent approach for translation with markup was the detag-and-project pipeline (Joanis et al., 2013; Müller, 2017; Zenkel et al., 2021). This pipeline usually leverages a machine translation (MT) system to translate plain text (with tags removed) and a separate word aligner to reinsert the tags into the translated text. Although straightforward, it is prone to error propagation from individual MT and alignment modules.

LLMs have emerged as a promising end-to-end solution for markup translation (Dabre et al., 2023, 2024). Few-shot prompting is a convenient way to enable LLMs to learn markup transfer patterns with only a few examples (Brown et al., 2020; Lewis et al., 2020; Dabre et al., 2023), and fine-tuning provides more robust domain adaptation capabilities thus better performance (Dabre et al., 2024). However, the training objective of supervised fine-tuning is to optimize token-level likelihood, leaving markup accuracy largely unaddressed. Therefore, it is difficult for them to handle complex structured documents such as the one shown in Figure 1.

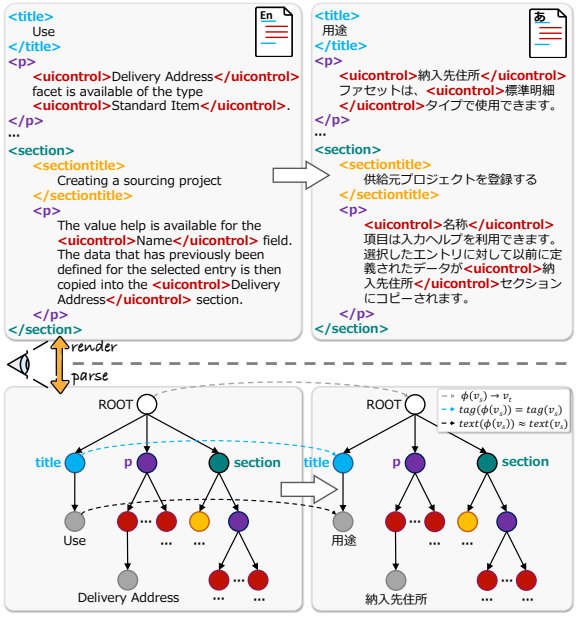


Figure 1: A structured document translation example (English→Japanese), with markup highlighted in color. The lower part shows the translation of XML tree structure with node $\phi(\cdot)$, tag$(\cdot)$, and text$(\cdot)$ mappings.

In this study, we address these limitations by proposing Format Reinforcement Learning (FORMATRL), which moves from the token-level likelihood optimization to directly optimizing structure-aware objectives. It first fine-tunes an LLM for basic document translation capability, then applies Group Relative Policy Optimization

[1]Our code is available at https://github.com/shyyhs/format-rl

Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 677–697
December 20-24, 2025

(GRPO) with two novel structure-aware rewards: TreeSim for measuring XML tree structural similarity via edit distance, and Node-chrF for node-level translation quality assessment. The main contributions of this paper are summarized below:

- We propose Format Reinforcement Learning (**FORMATRL**) for structured document translation. It utilizes Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to optimize structural fidelity through novel structure-aware rewards **TreeSim** and **Node-chrF**. We also investigate a range of additional rewards to reinforce structural fidelity and translation quality.
- We use a new metric, Structure-Aware Area Under Curve (**StrucAUC**), which distinguishes between minor errors and major failures, then robustly combines both translation and structural quality into a single score.
- Our experimental results demonstrate significant improvements on the software documentation dataset (Buschbeck and Exel, 2020) across four translation directions, with FORMATRL achieving average gains of 3.69 XML-Match, 2.16 XML-BLEU, 0.22 Content-BLEU, and 0.93 StrucAUC scores compared to a strong supervised fine-tuning baseline.

2 Related Work

This study focuses on §2.1 structured text translation and §2.2 reinforcement learning.

2.1 Structured Text Translation

The traditional detag-and-project approaches rely on separate modules for translation and markup handling (Du et al., 2010; Joanis et al., 2013; Müller, 2017; Hanneman and Dinu, 2020; Zenkel et al., 2021; Ryu et al., 2022; Steffen and van Genabith, 2021; Zenkel et al., 2021). However, these methods suffer from error propagation across modules, and the MT system translates at the sentence level without using document-level context.

Recent end-to-end approaches for structured text translation have become possible with LLMs such as BLOOM (Le Scao et al., 2022), ChatGPT (Brown et al., 2020; OpenAI, 2023), and Llama 3 (Dubey et al., 2024), owing to their strong in-context learning and generalization capabilities. Previous studies using few-shot prompting (Dabre et al., 2023) or fine-tuning on a small dataset (Dabre et al., 2024) work well for sentence translation with markup. However, they struggle to handle complex structures such as those found in XML documents.

2.2 Reinforcement Learning

From the perspective of RL, generation is a sequence of actions to maximize the reward. Proximal Policy Optimization (PPO) (Schulman et al., 2017) is the RL algorithm used in ChatGPT (OpenAI, 2023), whereas GRPO (Shao et al., 2024) used in DeepSeek-R1 (DeepSeek-AI, 2025) further simplifies PPO by removing the separate value network. RL has been applied to tasks such as code generation (Dou et al., 2024), JSON generation (Lu et al., 2025) and format instruction following (Yao et al., 2024). To our knowledge, we are the first to apply RL algorithms to the structured document translation task, whose challenge lies in designing rewards to guide generation of exactly the same structure as that in the source document while maintaining the high translation quality.

3 Method

Our pipeline is shown in Figure 2. We first define the task in §3.1, then describe the supervised fine-tuning (SFT) phase in §3.2, and finally present the core reinforcement learning phase in §3.3.

3.1 Task Definition

This work addresses the task of translating a structured document D_s in the source language into its counterpart D_t in the target language. A structured document D can be viewed as an XML tree $D = (V_D, E_D)$, where V_D denotes the set of nodes and E_D the set of parent–child edges. Each node is associated with a tag symbol $\mathit{tag}(v)$ (e.g., <p>) and may contain textual segments $\mathit{text}(v)$.

The translation model π_θ is a conditional probability distribution defined as follows:

$$\pi_\theta : \mathcal{D}_s \times \mathcal{D}_t \to [0,1] \subset \mathbb{R}, \quad \pi_\theta(D_t \mid D_s)$$

where $\pi_\theta(D_t \mid D_s)$ denotes the probability of generating the target document D_t given the source document D_s, and $\mathcal{D}_s$ and $\mathcal{D}_t$ are the spaces of all possible structured documents in the source and target languages. The predicted translation $\hat{D}_t$ is typically obtained by maximizing this probability:

$$\hat{D}_t = \arg\max_{D_t \in \mathcal{D}_t} \pi_\theta(D_t \mid D_s)$$

We assume that the predicted document $\hat{D}_t$ satisfies the following two conditions we target:

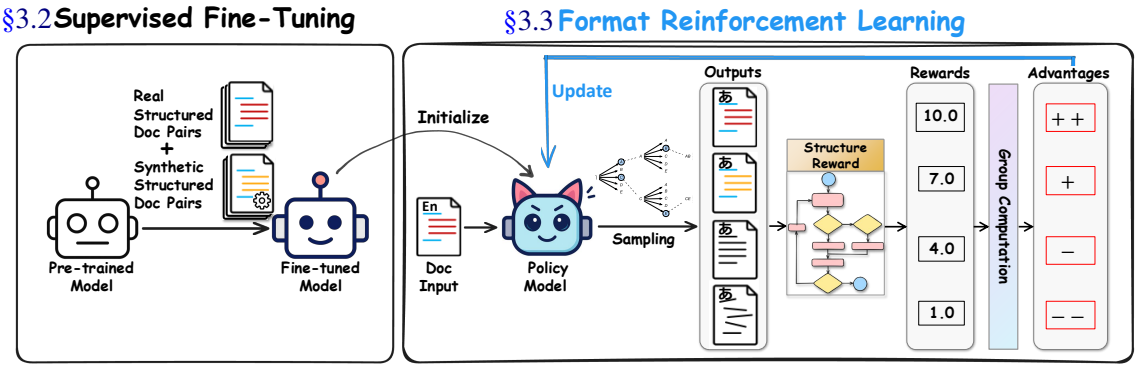


Figure 2: Our FORMATRL pipeline consists of two stages. First, we fine-tune a pre-trained LLM (e.g., Llama-3.1-8B-Instruct) using real and synthetic structured document pairs. Second, we reinforce the format handling ability through GRPO with our proposed format reward functions.

1. **Structural Identity**: $\hat{D}_t$ is isomorphic to the source tree D_s. Formally, there exists a bijection $\phi : V_{D_s} \to V_{\hat{D}_t}$ such that:
   - For any edge $(u, v) \in E_{D_s}$, we have $(\phi(u), \phi(v)) \in E_{\hat{D}_t}$.
   - For any internal node $v \in V_{D_s}$, the corresponding target node shares the same tag symbol: $tag(\phi(v)) = tag(v)$.
2. **Translation Correspondence**: For each source node $v \in V_{D_s}$ and its corresponding target node $\phi(v)$ their textual contents $text(v)$ and $text(\phi(v))$ are mutual translations.

To examine the extent to which both conditions are satisfied, in practice, we measure the translation quality between the predicted tree $\hat{D}_t$ and a reference document $D_t^\star$ using well-established metrics such as BLEU (Papineni et al., 2002) and COMET (Rei et al., 2020, 2022).

3.2 Phase I: Supervised Fine-Tuning

We fine-tune a pre-trained LLM on parallel structured documents. To address the data scarcity problem, we synthesize training data by injecting XML markup into parallel plain-text documents.

Data Synthesis. Given a parallel corpus of plain documents $\{(d_s^i, d_t^i)\}_{i=1}^N$, we use GPT-4o to generate structured documents $\{(D_s^i, D_t^i)\}_{i=1}^M$ in which:

- Both D_s^i and D_t^i have the same structure;
- The original parallel texts are preserved.

We ensure structural identity through validation: for each generated pair (D_s, D_t), we verify their XML trees are isomorphic. Invalid pairs are regenerated until success or hitting the retry limit.

3.3 Phase II: Format Reinforcement

Initialized from the SFT checkpoint, we use our designed rewards to optimize the translation model (policy model as termed in GRPO) to generate structurally correct and high-quality translations.

3.3.1 Reward Functions

The policy model learns from good samples generated by itself during training, where the reward function defines what is good. During GRPO training, a reward function $r(\hat{D}_{t,i}, D_t^\star)$ compares each sampled output $\hat{D}_{t,i} \sim \pi_\theta(\cdot|D_s)$ with the reference document $D_t^\star$, and indicates how good each output is. To reinforce structure-aware similarity, we propose two rewards: TreeSim and Node-chrF.

TreeSim measures structural similarity between the predicted and reference XML trees. It first parses both documents as XML fragments wrapped in a dummy root. The similarity is computed using the Zhang-Shasha tree edit distance (Zhang and Shasha, 1989), which counts the minimum number of node insertions, deletions, or relabelings needed to transform one tree into another. To obtain a normalized similarity score, we use:

$$\text{TreeSim}(\hat{D}_{t,i}, D_t^\star) = 1 - \frac{\text{EditDist}(\hat{D}_{t,i}, D_t^\star)}{\max(|\hat{D}_{t,i}|, |D_t^\star|)}$$

where EditDist is the tree edit distance and $|D|$ denotes the number of nodes in tree D excluding the dummy root. This normalization ensures that the score remains in $[0, 1]$, with 1 indicating identical structures and 0 maximum dissimilarity. Specifically, we assign a penalty score of -0.1 for invalid XML that cannot be parsed.

Node-chrF measures translation quality at the level of individual XML nodes. The algorithm performs a parallel depth-first traversal of the predicted and reference XML trees, pairing nodes at corresponding positions. When the sizes differ, the algorithm extends the shorter traversal list to match the longer one by adding empty placeholders. For each node pair $(v_{\text{pred}}, v_{\text{ref}})$, the metric computes:

- A score of 0 if the nodes have mismatched tags (e.g., `<p>` vs `<h1>`) or are unpaired (e.g., one subtree has more nodes than the other)
- The chrF score (Popović, 2015) of their textual content (excluding child nodes) if tags match
- Skip node pairs that contain only whitespace

The final score is the average of all node pairs:

$$\text{Node-chrF} = \frac{1}{|\mathcal{P}|} \sum_{(v_{\text{pred}}, v_{\text{ref}}) \in \mathcal{P}} \mathbb{1}_{\text{match}}(v_{\text{pred}}, v_{\text{ref}}) \cdot \text{chrF}(v_{\text{pred}}, v_{\text{ref}})$$

, where $\mathcal{P}$ is the set of all node pairs in the traversal and $\mathbb{1}$ is the indicator function for tag matching. For aligned trees, this metric focuses on translation quality. If the translation contains structural mistakes, however, nodes become misaligned, and the reward degrades substantially.

In practice, we scale each reward to $|r| \in [0, 10]$ for numerical stability. We also investigate the use of other metrics (§6.5) as rewards and explore combining two rewards (by summing the scores).

3.3.2 Optimization

After calculating reward scores for a group of samples, we encourage the model to generate similar high-scoring outputs. In GRPO, we calculate the relative performance comparisons within the group, called advantages, which is then used to update the document translation policy model π_θ.

Formally, the optimization process works as follows: for each source document D_s, we generate K candidate translations $\{\hat{D}_{t,i}\}_{i=1}^{K}$ from the current policy π_θ. Instead of requiring absolute quality assessments, GRPO computes advantages by comparing each generation's reward against the group mean, effectively learning which translations are better than average within the same context. Since we perform a single gradient update per exploration stage when computing gradients, we can remove the min and clip operation. This yields the following objective:

$$\mathcal{L}_{\text{GRPO}} = - \mathbb{E}_{D_s \sim \mathcal{D}, \{\hat{D}_{t,i}\}_{i=1}^{K} \sim \pi_\theta(\cdot|D_s)} \left[\frac{1}{K} \sum_{i=1}^{K} \hat{A}_i \log \pi_\theta(\hat{D}_{t,i}|D_s) \right] \quad \text{(1a)}$$

$$+ \beta \cdot D_{KL}(\pi_\theta || \pi_{\text{SFT}}) \quad \text{(1b)}$$

The first term (1a) encourages the model to increase the likelihood of generations with positive advantages and to decrease the likelihood of those with negative advantages, with $\hat{A}_i$ computed as:

$$\hat{A}_i = \frac{r(\hat{D}_{t,i}, D_t^\star) - \bar{r}}{\sigma_r}$$

$$\bar{r} = \frac{1}{K} \sum_{j=1}^{K} r(\hat{D}_{t,j}, D_t^\star)$$

$$\sigma_r = \sqrt{\frac{1}{K} \sum_{j=1}^{K} (r(\hat{D}_{t,j}, D_t^\star) - \bar{r})^2}$$

The second term (1b) is a Kullback-Leibler divergence regularizer that prevents the optimized policy π_θ from deviating too far from the supervised fine-tuned model π_{SFT} with β controlling its strength, thereby avoiding catastrophic forgetting.

4 Evaluation Metrics: StrucAUC

Previous studies on structured data translation apply the XML-BLEU metric (Hashimoto et al., 2019) as a combined score for both translation quality and structural fidelity. However, it results in a zero score on the document-level even with minor structural mismatch. To provide a more fine-grained evaluation, we propose StrucAUC that distinguishes between minor errors and major structural failures. In detail, it provides a translation quality evaluation by interpolating between two scores, Node-chrF and Optimal Node-chrF, to measure quality with different levels of error tolerance.

Optimal Node-chrF provides a way to measure Node-chrF for two documents with slightly different structures by node alignment. It represents each node by its entire subtree (including tags and descendants) and computes a cost matrix C, where C_{ij} is the chrF distance between the i-th node's subtree in $\hat{D}_t$ and the j-th node's subtree in the $D_t^\star$. Using the Hungarian algorithm (Kuhn, 1955), it solves the linear sum assignment problem to find the optimal one-to-one mapping $\mathcal{M}^*$ that minimizes the total distance:

$$\mathcal{M}^* = \arg\min_{\mathcal{M}} \sum_{(v_{\text{pred}}, v_{\text{ref}}) \in \mathcal{M}} C_{v_{\text{pred}}, v_{\text{ref}}}$$

The final score evaluates chrF on node-level textual content (excluding children) under this optimal alignment: Nodes that cannot be matched are scored 0, ensuring all nodes are accounted for.

StrucAUC then integrates structural tolerance through tree edit distance (Zhang and Shasha, 1989) into Optimal Node-chrF. For each document, we calculate the minimum number of edits required to transform the predicted tree into its optimally aligned version (as determined by $\mathcal{M}^*$), with tag mismatches counting as 0.5 edits. The metric then computes a curve at the corpus level: at each edit threshold $k \in \{0, 0.5, 1, ..., K\}$, documents requiring at most k edits contribute their Optimal Node-chrF score, while others contribute their regular Node-chrF score. The area under this curve from 0 to K yields StrucAUC@K, providing a smooth degradation from perfect structural alignment to increasing structural deviations. This makes StrucAUC robust for document-level evaluation: minor structural errors (e.g., a misplaced formatting tag) result in proportional score reductions rather than complete failure, while still rewarding structural fidelity. In our experiments, we report StrucAUC@5, allowing up to 5 structural edits before considering a document structurally misaligned. We provide the pseudo code in Appendix H.

5 Experimental Settings

This section describes our dataset, evaluation metrics, and implementation details of our method.

5.1 Dataset

We use the SAP software documentation dataset (Buschbeck and Exel, 2020) that contains parallel structured documents for language pairs including Japanese–English and Chinese–English translated by professional translators. Each language pair consists of 190 document pairs for testing, and an additional 195 document pairs, of which we use 100 for training and 95 for development. Each source–target document pair contains the same number of lines with a one-to-one, linear alignment, reflecting the property of this task that the page layout in different languages should be identical.

Statistics Documents in this dataset exhibit substantial structural variety. After converting documents into XML trees, each tree has an average depth of 7.11 ± 1.51 and contains 27.36 ± 25.28 nodes, with a median of 18 nodes per document, and an average of 14.62 text segments per document. Overall, it covers 58 unique XML tags.

Inline Markup Setting The dataset also provides a simplified version with only sentence-internal markup, as shown in Figure 3. We name it the *inline markup setup*. Different from the structured setup which preserves non-translatable nodes (e.g., `<source>In-App Help</source>` and metadata), the inline setup keeps only translatable spans with inline tags. Consequently the reference texts also differ between the two setups. We construct the data for both setups with the official SAP XSLT preprocessing scripts.

Use
<uicontrol>Delivery Address**</uicontrol>** facet is available of the type **<uicontrol>**Standard Item**</uicontrol>**.
Creating a sourcing project
The value help is available for the **<uicontrol>**Name**</uicontrol>** field. The data that has previously been defined for the selected entry is then copied into the **<uicontrol>**Delivery Address**</uicontrol>** section.

Figure 3: Inline markup version of the example in Fig.1.

5.2 Evaluation

We apply six evaluation metrics, including four from previous studies: Content-BLEU, XML-Validity, XML-Match, XML-BLEU, and two proposed metrics: Content-COMET and StrucAUC. We classify them into three categories: 1) Translation: these mainly measure translation quality, 2) Structure: the ones measure structure fidelity, and 3) Combined: the ones measure both.

Translation Content-BLEU is the BLEU for a document with all XML markup removed. We employ the SacreBLEU tool (Post, 2018) with language-specific tokenizers.[2] **Content-COMET** is based on the neural MT metric COMET-22 (Rei et al., 2022). The metric is applied to the document texts without XML markup, as COMET-22 was not trained on structured documents and therefore cannot be directly applied to such data.

Structure XML-Validity returns a binary score of one or zero whether the output D_t passed XML parsing. XML-Match is also binary indicating whether the XML trees of output D_t and reference $D_t^\star$ are exactly the same.

Combined The XML-BLEU metric (Hashimoto et al., 2019) is a combined score for both translation quality and structural fidelity. First, both

[2]e.g., signature for Japanese: "nrefs:1|case:lc|eff:no|tok:ja-mecab-0.996-IPA|smooth:exp|version:2.5.1"

translated and reference documents are split into text segments at XML tag boundaries. If a translation's XML-Match is true, the segments are paired for BLEU computation. Otherwise, the reference's segments are paired with empty strings, thereby penalizing structural errors. The metric is then computed on the corpus level across all segments of all documents. **StrucAUC** is also a combined metric as described in §4.

Additionally, we report empirical p-values from statistical significance testing using bootstrap resampling with $1,000$ trials.

5.3 Implementations

We report implementation details and hyperparameters selected based on our preliminary experiments.

5.3.1 Prompting Baseline

For the few-shot prompting baseline (Dabre et al., 2023), we evaluate pre-trained language models (GPT-4o and meta-llama/Llama-3.1-8B-Instruct) using in-context learning with $k \in \{0, 1, 2, 3, 4, 5\}$ full document pairs as exemplars, and report the k-shot setting with the highest XML-BLEU. We use greedy decoding for deterministic outputs and set the maximum generation length to 2,000 tokens which is sufficient to accommodate long document translations.

5.3.2 Supervised Fine-Tuning

This section describes the LLM fine-tuning method as in Dabre et al. (2024) on our document-level data. We use Llama-3.1-8B-Instruct (Dubey et al., 2024) as the base model in our experiments.

Synthetic Data We use GPT-4o[3] to synthesize markup using the Asian Language Treebank (ALT) corpus (Thu et al., 2016; Riza et al., 2016), generating 900 structured document pairs per language. ALT contains high-quality general domain parallel document units and aligns with our target language pairs. The prompt includes a random example from the development set of SAP dataset and a sample of five tags from the target tag vocabulary. Without domain-specific guidance, LLM defaults to generic tags (e.g., `<person>`), causing train-test mismatches. See full prompt in Appendix A.

Hyperparameters The SFT model π_θ^{SFT} is then trained on both 100 real and 0 to 400 synthetic structured document pairs using standard cross-entropy loss. We fine-tune for 20 epochs with batch size of 8, a learning rate of 3×10^{-7} and cosine learning rate scheduling with a warmup ratio of 0.1. We use the AdamW optimizer (Loshchilov and Hutter, 2017). Early stopping is triggered after 10 evaluations without improvement, with evaluation performed every 10 steps.

5.3.3 Format Reinforcement

We now describe the hyperparameter configuration for the reinforcement learning phase (§3.3), chosen based on our preliminary experiments.

Training Configuration. We report results using TreeSim reward in §6.1, and results for Node-chrF and other rewards in §6.5. We use a small learning rate of 10^{-6} and train for 5 epochs with early stopping based on validation loss. Early stopping is triggered after 3 evaluation steps without improvement, with evaluation and checkpointing performed every 3 training steps. We set the maximum sequence length to $2,000$ tokens for both prompts and completions. The KL penalty coefficient β is set to the default value of 0.01. We select the checkpoint for testing based on the development set performance.

Batch and Generation Settings. We use 8 generations per document ($K = 8$) with a per-device batch size of 8 and gradient accumulation steps of 1, resulting in an effective batch size of 64 across 8 H200 GPUs. For generation, we use sampling with default temperature of 1.0.

5.3.4 Computational Efficiency

During training, we leverage DeepSpeed ZeRO-3 optimization and mixed precision training with bfloat16 for memory and computational efficiency. Each SFT model takes about 2.1 hours and GRPO model takes about 1.3 hours of training. We employ vLLM (Kwon et al., 2023) for efficient inference where it takes 2 minutes on test set.

6 Results and Analysis

6.1 Main Results

Table 1 presents our main results on the structured document translation task across four language pairs. FORMATRL using TreeSim reward consistently outperforms both the prompting and SFT baselines across nearly all evaluation metrics. Results of other rewards are shown in Appendix C.

Structural Fidelity Improvements. FORMATRL with TreeSim shows significant gains in structural preservation. XML-Match scores improve by an

[3] gpt-4o-2024-08-06

Src→Tgt	Method	Translation		Structure		Combined		Avg.
		Content-BLEU	Content-COMET	XML-Validity	XML-Match	XML-BLEU	StrucAUC	
En→Zh	Prompt	49.88	86.16	91.05	76.84	27.50	57.75	64.86
	SFT	49.66	86.47	94.21	85.26	36.38	63.57	69.26
	FORMATRL	**49.88**	**86.48**	**95.26**	**87.37**	**38.07**	**64.12**	**70.20**
Zh→En	Prompt	48.82	85.25	93.16	82.11	26.34	71.39	67.84
	SFT	**56.41**	**85.34**	94.74	83.68	27.58	71.66	69.90
	FORMATRL	56.28	85.25	**95.26**	**86.84**	**29.14**	**72.84**	**70.94**
En→Ja	Prompt	36.60	87.17	87.89	67.37	14.49	48.60	57.02
	SFT	39.11	**88.22**	95.26	84.21	27.47	60.40	65.78
	FORMATRL	**39.30**	88.20	**95.79**	**88.42**	**30.32**	**60.48**	**67.09**
Ja→En	Prompt	44.14	85.93	90.53	80.00	22.38	65.25	64.70
	SFT	52.19	85.96	**95.26**	82.11	24.15	67.92	67.93
	FORMATRL	**52.79**	**86.01**	94.74	**87.37**	**26.67**	**69.82**	**69.57**

Table 1: Results of FORMATRL and two baselines on structured documents. Bold indicates **the best performance**. Background colors indicate statistical significance $p < 0.05$ compared to SFT.

Src→Tgt	Method	Translation		Structure		Combined		Avg.
		Content-BLEU	Content-COMET	XML-Validity	XML-Match	XML-BLEU	StrucAUC	
En→Zh	Prompt	54.79	85.87	96.32	84.74	43.33	56.62	70.28
	SFT	**57.95**	86.19	**98.42**	89.47	47.51	63.14	73.78
	FORMATRL	57.70	**86.22**	**98.42**	**90.00**	**47.63**	**64.29**	**74.04**
Zh→En	Prompt	**39.92**	**83.31**	95.26	83.68	33.83	67.06	**67.18**
	SFT	32.24	83.06	95.26	81.05	33.01	65.77	65.07
	FORMATRL	34.76	82.83	**95.79**	**84.74**	**34.74**	**66.28**	66.52
En→Ja	Prompt	40.13	87.90	96.32	79.47	26.78	45.07	62.61
	SFT	44.42	88.40	97.37	84.21	32.27	54.93	66.93
	FORMATRL	**45.60**	**88.60**	**98.42**	**86.84**	**35.44**	**55.04**	**68.32**
Ja→En	Prompt	35.61	84.86	**98.95**	81.58	27.58	64.65	65.54
	SFT	34.74	84.66	97.89	82.63	26.72	63.60	65.04
	FORMATRL	**37.02**	**84.96**	98.42	**86.84**	**30.13**	**65.82**	**67.20**

Table 2: Results of FORMATRL and two baselines on inline markup dataset. Bold indicates **the best performance**. Background colors indicate statistical significance $p < 0.05$ compared to SFT.

average of 3.69 over SFT, with the largest improvement of 5.26 points observed for Ja→En. This indicates that FORMATRL effectively learns to maintain document structure beyond what SFT achieves.

Translation Quality Gains. Importantly, FORMATRL maintains or slightly improves translation quality while enhancing structural fidelity. Content-BLEU scores increase by an average of 0.22 points over SFT. Content-COMET scores remain stable, suggesting that our structural improvements do not come at the cost of translation quality.

Combined Performance. We show the combined improvement through XML-BLEU, which is widely used in previous work on structured data translation (Hashimoto et al., 2019; Dabre et al., 2024). It improves by 2.16 points on average, and our proposed StrucAUC metric shows gains of 0.93 points, confirming that improvements are robust across different structural error tolerances.

Human Evaluation. We performed a small-scale human evaluation on 60 rendered En→Ja pages comparing FORMATRL and prompting methods. For each page, an annotator compared the outputs of two methods against the reference, where the order of two outputs are randomly shuffled each time. Results show FORMATRL won 29, prompting won 13, and 18 were ties. Qualitatively, outputs with (i) correct structure and (ii) correct embedded UI were preferred, suggesting that structural fidelity may be important in user experience.

6.2 Results on Documents with Inline Markup

Table 2 presents results on the inline markup dataset, where structural complexity is reduced to inline markup. We found that although FORMATRL with TreeSim still shows improvements in all metrics, the performance gap between the baseline method and FORMATRL narrows considerably compared to structured documents. For ex-

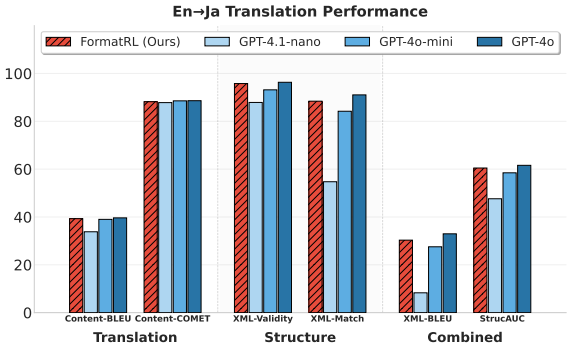


Figure 4: Comparison with GPT-4.1-nano (2025-04-14), GPT-4o-mini (2024-07-18), and GPT-4o (2024-08-06).

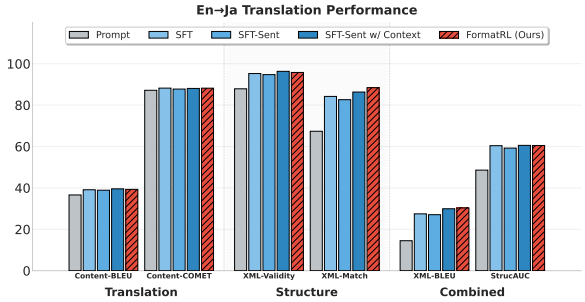


Figure 5: Comparison with parse-and-assemble baselines, in which the LLM acts as sentence-level MT model with or without document context.

ample, the XML-Match gap between Prompt and FormatRL narrows from 10.92 to 4.74. This suggests that LLMs handle simpler inline structures effectively through in-context learning, but struggle with more structured documents. We show results of different rewards in Appendix D.

6.3 Comparison with GPT-4 Models

We compare our approach to three GPT models which serve as reference. We show results of En→Ja in Figure 4 and all directions in Appendix F. We found FormatRL shows comparable performance on most metrics with GPT-4o and outperforms GPT-4.1-nano and GPT-4o-mini. Although with similar scores in automatic evaluation, after analyzing 60 outputs of GPT-4o and our model, we found FormatRL outputs match the style (e.g. word choice is more formal) in source documents better than prompting with GPT-4o.

6.4 Comparison with Parse-and-Assemble

We implemented two parse-and-assemble baselines, where we first extract translatable text blocks, then apply an LLM-based sentence-level translator, and finally assemble the texts to form the output document. SFT-Sent trains Llama 3.1 8B on parallel sentences whereas SFT-Sent w/ Content extents this by providing the whole document as context. Figure 5 shows that for En→Ja, translation quality is comparable but FormatRL achieves higher XML-Match. Although parse-and-assemble ensures correct document structure, it struggles with in-line tags whose positions vary across target language syntax. Furthermore, providing full documents for every sentence makes training 4.2× slower and inference 5.7× slower than standard SFT, showing that the end-to-end paradigm offers a more natural and efficient solution.

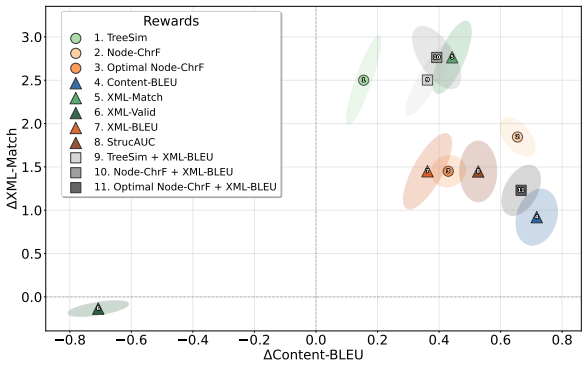


Figure 6: Improvement of FormatRL over SFT using various single rewards, and combinations of two rewards. Points represent mean improvement and ellipses visualize the local covariance directional structure between two metrics improvements.

6.5 Analysis: Reward Choice

Figure 6 shows the effect of different reward functions during GRPO training, including: 1) proposed TreeSim and Node-chrF, 2) metrics used in evaluation as rewards,[4] and 3) combination of two rewards. Estimates are constructed from 8 runs of RL results. Refer to Appendix G for results featuring COMET instead of BLEU.

First, we found all rewards except XML-Validity to improve translation quality measured by Content-BLEU. Even pure structure-aware rewards, such as TreeSim and XML-Match, can improve translation. The combined reward Node-chrf improves both in good balance. However, not aligning to the reference (XML-Validity) is bad, hurting both translation and structure quality. Second, the best way to optimize a specific metric is using it as a reward. Reinforcement learning with Content-BLEU as reward achieves the highest gain in Content-BLEU, and similarly, the XML-Match reward achieves the best XML-Match performance. Finally, we observe

[4]Content-BLEU and XML-BLEU here are document-level.

Src→Tgt	Method	TreeSim	Node-chrF
	Prompt	95.97	56.33
	SFT	97.86	62.67
En→Zh	FORMATRL		
	w/ TreeSim	**98.19**	63.24
	w/ Node-chrF	97.70	**64.41**
	Prompt	97.24	69.97
	SFT	97.54	70.58
Zh→En	FORMATRL		
	w/ TreeSim	**98.12**	71.77
	w/ Node-chrF	97.97	**73.00**
	Prompt	94.40	47.64
	SFT	97.55	**59.77**
En→Ja	FORMATRL		
	w/ TreeSim	**97.98**	**59.77**
	w/ Node-chrF	97.23	59.26
	Prompt	96.69	63.93
	SFT	97.82	66.47
Ja→En	FORMATRL		
	w/ TreeSim	**98.28**	68.51
	w/ Node-chrF	98.02	**68.93**

Table 3: Performance comparison when optimizing TreeSim and Node-chrF rewards. Bold indicates the best performance.

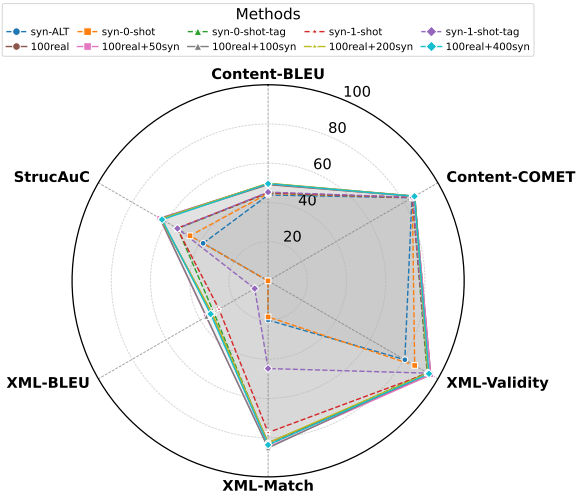


Figure 7: SFT Model performance comparison for English to Chinese translation by composition of training data. **syn-ALT** refers to fine-tuning using the raw ALT document pairs. **syn-0-shot** refers to data synthesized in a zero-shot manner. For **syn-1-shot**, the synthesizing LLM was provided with one example. The ***-tag** setups additionally guided the LLM with example XML tags from the development set. The **Xreal+Ysyn** setups are a mixture of real data and synthetic data generated with the syn-1-shot-tag approach.

reward combination yields averaging effects, e.g., combining TreeSim with XML-BLEU shows better Content-BLEU improvement than TreeSim alone.

6.6 Analysis: Reward-Metric Alignment

Table 3 compares the direct optimization effects of our two proposed rewards. We observe clear reward-metric alignment: using TreeSim as reward achieves the highest TreeSim scores, while Node-chrF reward yields the best Node-chrF scores in most directions. This confirms that reinforcement learning can effectively improve the specific structural properties defined by the reward.

6.7 Analysis: Synthetic Data Strategies

We explore the effect of using different synthetic data strategies to train the SFT model. As shown in Figure 7, although the translation quality comes close to training on real data, using synthetic data alone can lead to catastrophic structure failure, with XML-Match scores dropping below 20%. We suppose that the domain shift in textual content likely has adverse interaction effects with the structural performance. Because it is unlikely the model completely independently learns structural transfer and translation. This phenomenon highlights the crucial role of real target-domain XML markup. The importance of such markup is further underscored by observations that models trained on synthetic data generated without explicit guidance from in-domain examples and markup tags were more prone to structural errors. When combined with real data using the syn-1-shot-tag synthetic data, moderate amounts of synthetic data (e.g., 100real+100syn) can improve performance, whereas excessive amounts (e.g., 100real+400syn) can degrade it. For translation quality, this is not surprising: the Asian Language Treebank (Thu et al., 2016) used for data generation differs substantially in domain from software documentation. Results of all language pairs are shown in Figure 10.

7 Conclusion

To address the challenge of translating documents with complex structures, we propose FORMATRL, a novel reinforcement learning approach with proposed structure-aware rewards: TreeSim and Node-chrF. We further propose StrucAUC as a fine-grained evaluation metric. Experimental results show FORMATRL improves the structural fidelity of translated documents without compromising translation quality across both simple inline markup and complex structured documents.

8 Limitations

Limited Tag Set. We restricted the tag set used during synthetic data generation to those present in the development set. While this approach provides consistency, it raises questions about the downstream translation models' ability to extrapolate to documents containing previously unseen tags. We did not evaluate this extrapolation capability due to budget constraints, as such an experiment would require generating substantially larger quantities of synthetic data with diverse markup using GPT. We did not explore tag abstraction using placeholder tags (e.g., `<t1>`) as which may help generalization but in the same time introduces pre-/post-editing and ignore semantics in human-interpretable tags.

Applying Sentence-level Metrics to Documents. While we applied BLEU and COMET-22 to XML-stripped documents, these metrics, however, have known shortcomings when applied at the document-level as they are not designed/trained for such data (Jiang et al., 2022; Vernikos et al., 2022).

Lack of Rigorous Human Evaluation. We performed a simple human evaluation in the result section. However, we are aware that a rigorous evaluation would require multiple annotators together with well-defined annotation instructions such as error taxonomies tailored to structured documents (e.g., MQM (Freitag et al., 2021) or ESA (Kocmi et al., 2024) style annotation) that explicitly capture tag mismatch, nesting errors, and their severities. We leave this to future work.

Acknowledgements

We would like to thank the reviewers for their insightful comments and suggestions. This work was supported by JSPS KAKENHI Grant-in-Aid for Early-Career Scientists 25K21290.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Bianka Buschbeck and Miriam Exel. 2020. A parallel evaluation data set of software documentation with document structure annotation. In *Proceedings of the 7th Workshop on Asian Translation*, pages 160–169.

Raj Dabre, Bianka Buschbeck, Miriam Exel, and Hideki Tanaka. 2023. A study on the effectiveness of large language models for translation with markup. In *Proceedings of Machine Translation Summit XIX, Vol. 1: Research Track*, pages 148–159, Macau SAR, China. Asia-Pacific Association for Machine Translation.

Raj Dabre, Haiyue Song, Miriam Exel, Bianka Buschbeck, Johannes Eschbach-Dymanus, and Hideki Tanaka. 2024. How effective is synthetic data and instruction fine-tuning for translation with markup using LLMs? In *Proceedings of the 16th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 73–87, Chicago, USA. Association for Machine Translation in the Americas.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv*.

Shihan Dou, Yan Liu, Haoxiang Jia, Enyu Zhou, Limao Xiong, Junjie Shan, Caishuang Huang, Xiao Wang, Xiaoran Fan, Zhiheng Xi, Yuhao Zhou, Tao Ji, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024. StepCoder: Improving code generation with reinforcement learning from compiler feedback. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4571–4585, Bangkok, Thailand. Association for Computational Linguistics.

Jinhua Du, Johann Roturier, and Andy Way. 2010. TMX markup: A challenge when adapting SMT to the localisation environment. In *Proceedings of the 14th Annual Conference of the European Association for Machine Translation*, Saint Raphaël, France. European Association for Machine Translation.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, and 1 others. 2024. The llama 3 herd of models. *arXiv*.

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.

Greg Hanneman and Georgiana Dinu. 2020. How should markup tags be translated? In *Proceedings of the Fifth Conference on Machine Translation*, pages 1160–1173, Online. Association for Computational Linguistics.

Kazuma Hashimoto, Raffaella Buschiazzo, James Bradbury, Teresa Marshall, Richard Socher, and Caiming Xiong. 2019. A high-quality multilingual dataset for structured documentation translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 116–127, Florence, Italy. Association for Computational Linguistics.

Yuchen Jiang, Tianyu Liu, Shuming Ma, Dongdong Zhang, Jian Yang, Haoyang Huang, Rico Sennrich, Ryan Cotterell, Mrinmaya Sachan, and Ming Zhou. 2022. BlonDe: An automatic evaluation metric for document-level machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1550–1565, Seattle, United States. Association for Computational Linguistics.

Eric Joanis, Darlene Stewart, Samuel Larkin, and Roland Kuhn. 2013. Transferring markup tags in statistical machine translation: a two-stream approach. In *Proceedings of the 2nd Workshop on Post-editing Technology and Practice*, Nice, France.

Tom Kocmi, Vilém Zouhar, Eleftherios Avramidis, Roman Grundkiewicz, Marzena Karpinska, Maja Popović, Mrinmaya Sachan, and Mariya Shmatova. 2024. Error span annotation: A balanced approach for human evaluation of machine translation. In *Proceedings of the Ninth Conference on Machine Translation*, pages 1440–1453, Miami, Florida, USA. Association for Computational Linguistics.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

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

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, and 1 others. 2022. BLOOM: A 176b-parameter open-access multilingual language model. *arXiv*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Yaxi Lu, Haolun Li, Xin Cong, Zhong Zhang, Yesai Wu, Yankai Lin, Zhiyuan Liu, Fangming Liu, and Maosong Sun. 2025. Learning to generate structured output with schema reinforcement learning. *arXiv*.

Mathias Müller. 2017. Treatment of markup in statistical machine translation. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 36–46, Copenhagen, Denmark. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *arXiv*.

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.

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

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.

Hammam Riza, Michael Purwoadi, Gunarso, Teduh Uliniansyah, Aw Ai Ti, Sharifah Mahani Aljunied, Luong Chi Mai, Vu Tat Thang, Nguyen Phuong Thai, Vichet Chea, Rapid Sun, Sethserey Sam, Sopheap Seng, Khin Mar Soe, Khin Thandar Nwet, Masao Utiyama, and Chenchen Ding. 2016. Introduction of the asian language treebank. In *2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA)*, pages 1–6.

Yonghyun Ryu, Yoonjung Choi, and Sangha Kim. 2022. Data augmentation for inline tag-aware neural machine translation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 886–894, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv*.

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*.

Jörg Steffen and Josef van Genabith. 2021. TransIns: Document translation with markup reinsertion. In

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 28–34, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ye Kyaw Thu, Win Pa Pa, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Introducing the Asian language treebank (ALT). In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1574–1578, Portorož, Slovenia. European Language Resources Association (ELRA).

Giorgos Vernikos, Brian Thompson, Prashant Mathur, and Marcello Federico. 2022. Embarrassingly easy document-level MT metrics: How to convert any pretrained metric into a document-level metric. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 118–128, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Jiashu Yao, Heyan Huang, Zeming Liu, Haoyu Wen, Wei Su, Boao Qian, and Yuhang Guo. 2024. Reff: Reinforcing format faithfulness in language models across varied tasks. *arXiv*.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2021. Automatic bilingual markup transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3524–3533, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kaizhong Zhang and Dennis Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18:1245–1262.

A Synthetic Data Generation

We show the prompt template used for synthetic data generation in Figure 8. It instructs GPT-4o to augment existing translation pairs without markup by inserting hierarchical XML markup elements into both source and target documents while maintaining alignment between the structures. In all k-shot settings we include full source–target document pairs as exemplars. For k=5, their combined length is about ∼9,800 characters (∼5,070 Llama 3.1 tokens), which will differ for different language pairs.

B Example: Metrics Calculation

We illustrate how the metrics work using a toy example in Figure 9, including XML-Validity, XML-Match, XML-BLEU, and the proposed StrucAUC (including the calculation of Node-chrF and Optimal Node-chrF).

C Full Results of Structured Document Setting

Table 4 shows the detailed results of each reward function on the structured document setting across four language pairs (En→ Zh, Zh→ En, En→ Ja, Ja→ En). We found most of the rewards except XML-Validity improves across most metrics compared to supervised fine-tuning. Among the single reward functions, TreeSim usually achieves the best on structure score XML-Match, while Node-chrF shows the highest combined scores in three directions.

D Full Results of Inline Markup Setting

Table 5 shows the detailed results of each reward function on the inline markup setting across four language pairs (En→ Zh, Zh→ En, En→ Ja, Ja→ En). We have similar observations that most of the rewards except XML-Validity improves across most metrics compared to supervised fine-tuning especially on structure and combined scores.

E Synthetic Data Performance: All Translation Directions

We show the effect of using different data when training the SFT model for all translation directions in Figure 10. The trends are similar: involving real data usually surpass pure synthetic data by a large margin (which is also expected). And the ratio of synthetic data and real data did not affect the performance that much. Usually 1:1 is a good balance, where too much synthetic data such as 4:1 slightly hurts the performance of the SFT model trained on such data.

F Comparison with GPT

We compare our approach to three GPT variants of different sizes in Figure 11 which contains full results across all translation directions. FORMATRL is usually better than GPT-4.1-nano, comparable to GPT-4o-mini, and not as good as GPT-4o. We found GPT-4o is especially strong at preserving XML markup, achieving the highest scores on structural and combined metrics.

G Ablation: Reward Choice on Content-COMET and XML-Match

Similar to Figure 6, Figure 12 shows the effect of different reward functions during GRPO training using the improvement of Content-COMET instead of that of Content-BLEU. We found this time using Content-BLEU as a reward function did not achieve the best improvement on Content-COMET, indicating the effect of reward overfitting: achieving the best on a given metric while it is promised to achieve the best on a similar metric (that measures the similar dimension). Nevertheless, TreeSim achieved the highest XML-Match improvement among single rewards except using XML-Match itself, and Node-chrF achieved the highest Content-COMET improvement, indicating the effect of the proposed rewards.

H Details in StrucAUC

We show the StrucAUC algorithm in Algorithm 1. It is a very fast algorithm, the Hungarian matching is $O(n^3)$ in the number of text nodes n per document, where in our dataset n<20, so matching costs are negligible versus model inference. Tree edit operations are linear in tree size under our restricted XML grammar.

I Discussion

We discuss some interesting aspects we think of our implementation for people who are interested in these details.

About the dataset, we wanted to try FORMATRL on multiple datasets, after searching extensively for structured document translation datasets, we only found the SAP software document dataset. In the future we plan to curate some by ourselves.

Synthetic Data Generation Prompt

Your task is to synthesize training data for machine translation of structured XML documents. Given a provided translation pair, insert well-aligned hierarchical XML markup into both source and target document. Do not translate the markup elements or include language codes. Here is an example of a well-aligned document pair:

SOURCE:
<!DOCTYPE concept PUBLIC "-//SAP//DTD SAP DITA Composite//EN" "sap-ditabase.dtd">
<concept id="loio16f62395d57f487e9937a092e4caefe9" xml:lang="en-US">
<title>
Download
</title>
<shortdesc>
Downloads G/L account mappings into a .CSV file.
...

TARGET:
<!DOCTYPE concept PUBLIC "-//SAP//DTD SAP DITA Composite//EN" "sap-ditabase.dtd">
<concept id="loio16f62395d57f487e9937a092e4caefe9" xml:lang="en-US">
<title>
ダウンロード
</title>
<shortdesc>
G/L 勘定マッピングを.CSV ファイルにダウンロードします。 file.
...

Now, insert markup into the following document pair. Output only the augmented source and target documents. Here are some example markup tags:
<source></source>
<uicontrol></uicontrol>

<prolog></prolog>

SOURCE:
Every year around November 5th, people in Great Britain and some parts of the Commonwealth celebrate Guy Fawkes ...

TARGET:
毎年11月5日前後に、グレートブリテンと連邦の一部地域の人々は、１６０５年１１月５日に国会議事堂を爆破することができなかったヨーク...

Figure 8: Prompt template of **syn-1-shot-tag** used for data synthesis for the *structured markup* translation task. This prompt features an example document pair from the development set as well as example tags sampled from development data to guide the LLM in data synthesis. For the **syn-1-shot** setup, the example tags are withhold. For the **syn-0-shot-tag**, the one-shot example is withhold. **syn-0-shot** features only the initial prompt and the data to synthesize from. For the *inline* setup, the initial prompt is altered to 'Your task is to synthesize training data for machine translation of documents containing XML markup. Given a provided translation pair, insert well-aligned XML markup into both source and target document. Here is an example of a well-aligned document pair'.

About hyper-parameters, we found GRPO does not require much training signal is the base SFT model has the basic structured document translation ability. In this case, the learning rate is a crucial parameter, we have tried learning rate from 1e-5 to 1e-7 and found 1e-6 is a good balance. Additionally, we save the checkpoint and evaluate it every 3 steps to capture the best one. Due to its efficiency, each training takes no more than 1.5 hours and we in total spend less than 800 GPU hours (100 hours in 8 H200 GPUs) for all GRPO experiments. For the memory efficiency, we found setting $K = 8$, $BatchSize = 8$, and max generation token of 800 fits one H200 GPU with 141GB memory.

In our analysis, we used Content-BLEU and XML-Match as reward, which may sounds like overfitting the metrics. However, the KL regularizer is added in the loss which prevents degenerate solutions that optimize only a single metric. Moreover, **our proposed novel metrics TreeSim and Node-chrF do not overfit any metrics used in evaluation**.

Model Translation

```
<!DOCTYPE concept PUBLIC "sap-database.dtd">
<concept id="loio7683f14884a14e508">
  <title>
    Type of address.
  </title>
  <conbody>
    <prolog>
      <source>In-App Assistance</source>
    </prolog>
    <section>
      <p>
        Select the address type (<uicontrol>Company</uicontrol>
        Address for a safety data sheet) from the input assistance.
      </p>
    </section>
  </conbody>
</concept>
```

Reference

```
<!DOCTYPE concept PUBLIC "sap-database.dtd">
<concept id="loio76839b033ef14884a14e65a211a2d508">
  <title>
    Address Type
  </title>
  <prolog>
    <source>In-App Help</source>
  </prolog>
  <conbody>
    <section>
      <title>Use</title>
      <p>
        Select the address type (<uicontrol>Company Address</uicontrol>
        for a safety data sheet) from the input help.
      </p>
    </section>
  </conbody>
</concept>
```

Figure 9: Toy-example of a model translation and reference with markings for purely structural errors.
XML-Validity: The translation can be successfully parsed into an XML and therefore achieves a score of 1.
XML-Match: The translation does not match the exact structure of the reference and therefore scores 0.
XML-BLEU: Since the translation XML tree does not match the one of the reference the node contents of the reference will be paired with empty translations - e.g. (" ", "In-App-Help") - for corpus BLEU computation.
StrucAUC: The score is computed as a corpus level area under curve based on the respective *Node-chrF* and *Optimal Node-chrF*.
Node-chrF: The structural errors of the translation will lead to misalignment in the parallel depth-first traversal. For instance, we will see a pairing of [...,(<conbody>,<prolog>), (<prolog>, <source>), (<source>, <conbody>)...] which overall results in a low node-level chrF score of 16.89.
Optimal Node-chrF: With 3.5 edit operations (note that changing the label of the <concept> is considered half an edit), the nodes of the translation can be realigned to the reference, resulting in a Optimal Node-chrF of 52.92.

J License

We use the SAP software documentation dataset which is under **The Creative Commons license Attribution-Non Commercial 4.0 International (CC BY-NC 4.0)**, Asian Language Treebank (ALT) corpus under **The Creative Commons Attribution 4.0 International (CC BY 4.0) License**, and pre-trained models such as Llama-3.1-8B-Instruct under **The Llama 3.1 Community License** for research, which is consistent with their intended use. We have verified that the datasets do not contain personal information or offensive content.

We plan to release our code upon acceptance under **The Creative Commons license Attribution-Non Commercial 4.0 International (CC BY-NC 4.0)**. The code is intended for research purposes only and may not be used for commercial applications without explicit permission.

K The Use of AI Assistants

We used AI assistants for grammar and spelling checks. We sometimes also turn our incoherent listings of thoughts into a coherent paragraph which has always undergone further manual revisions.

Algorithm 1: StrucAUC Metric

Input: Hypotheses $\{\hat{D}_{t,i}\}_{i=1}^{n}$, References $\{D_t^{\star,i}\}_{i=1}^{n}$, Maximum operations K
Output: StrucAUC score

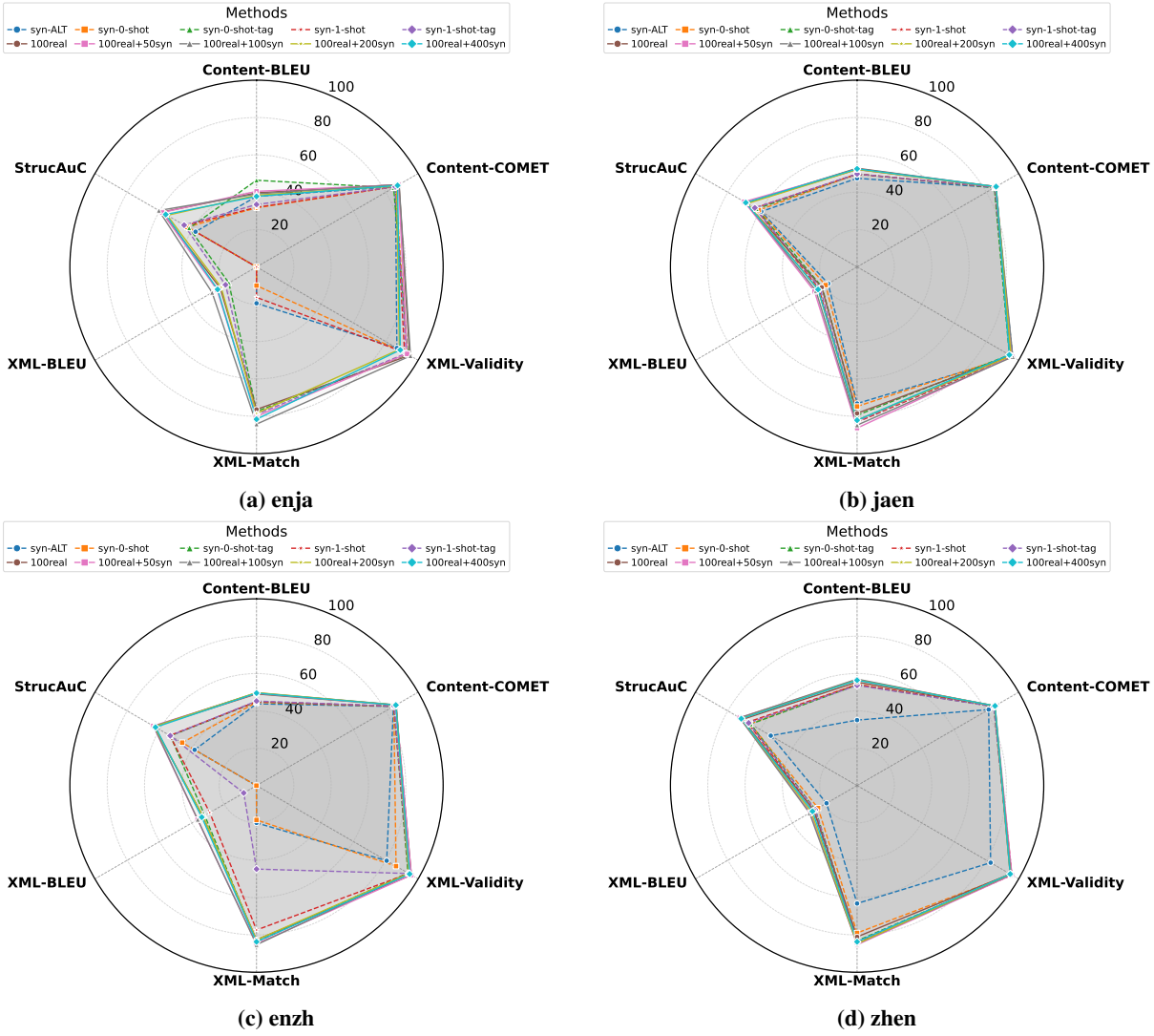
Initialize $S_k \leftarrow \{\}$ for $k \in \{0, 0.5, 1, \ldots, K\}$;
for $i = 1$ **to** n **do**
    Parse $\hat{D}_{t,i}$ and $D_t^{\star,i}$ to XML trees;
    if $D_t^{\star,i}$ *invalid* **then**
        continue;
    if $\hat{D}_{t,i}$ *invalid* **then**
        Add 0 to all S_k and **continue**;
    $s_{\text{unaligned}} \leftarrow \text{Node-chrF}_{\text{parallel}}(\hat{D}_{t,i}, D_t^{\star,i})$;
    $\mathcal{M}^* \leftarrow \text{OptimalAlignment}(\hat{D}_{t,i}, D_t^{\star,i})$ `// Hungarian algorithm`
    $d \leftarrow \text{TreeEditDistance}(\hat{D}_{t,i}, D_t^{\star,i}, \mathcal{M}^*)$;
    $s_{\text{optimal}} \leftarrow \text{Node-chrF}_{\text{optimal}}(\mathcal{M}^*)$;
    $S_0 \leftarrow S_0 \cup \{s_{\text{unaligned}}\}$;
    for $k \in \{0.5, 1, \ldots, K\}$ **do**
        if $d \leq k$ **then**
            $S_k \leftarrow S_k \cup \{s_{\text{optimal}}\}$;
        else
            $S_k \leftarrow S_k \cup \{s_{\text{unaligned}}\}$;
Compute AUC via trapezoidal integration over $\{(k/K, \text{mean}(S_k))\}$;
return AUC $\times 100$;

Src→Tgt	Method	Translation		Structure		Combined	
		Content-BLEU	Content-COMET	XML-Validity	XML-Match	XML-BLEU	StrucAUC
En→Zh	Prompt	49.88	86.16	91.05	76.84	27.50	57.75
	SFT	49.66	86.47	94.21	85.26	36.38	63.57
	FORMATRL *w/ Reward of:*						
	TreeSim	49.88	86.48	95.26	87.37	38.07	64.12
	Node-chrF	50.07	86.54	95.26	86.32	38.31	65.39
	Node-chrF (opt.)	49.70	86.47	94.74	85.26	36.17	63.06
	Content-BLEU	49.78	86.32	94.74	85.79	36.64	64.01
	XML-Validity	49.31	86.21	95.26	85.26	36.17	63.65
	XML-Match	49.75	86.37	95.79	85.26	35.97	64.20
	XML-BLEU	49.38	86.41	95.26	84.74	35.56	63.29
Zh→En	Prompt	48.82	85.25	93.16	82.11	26.34	71.39
	SFT	56.41	85.34	94.74	83.68	27.58	71.66
	FORMATRL *w/ Reward of:*						
	TreeSim	56.28	85.25	95.26	86.84	29.14	72.84
	Node-chrF	57.34	85.36	95.79	87.89	31.22	74.12
	Node-chrF (opt.)	56.98	85.32	95.26	86.84	30.52	72.76
	Content-BLEU	57.71	85.41	95.26	85.26	30.34	72.75
	XML-Validity	55.94	85.16	94.74	86.32	28.73	71.42
	XML-Match	56.39	85.16	94.74	87.89	30.31	72.33
	XML-BLEU	57.35	85.49	94.74	87.37	30.17	73.56
En→Ja	Prompt	36.60	87.17	87.89	67.37	14.49	48.60
	SFT	39.11	88.22	95.26	84.21	27.47	60.40
	FORMATRL *w/ Reward of:*						
	TreeSim	39.30	88.20	95.79	88.42	30.32	60.48
	Node-chrF	39.56	88.07	95.79	81.58	26.12	60.29
	Node-chrF (opt.)	39.38	88.19	94.21	83.16	26.46	59.52
	Content-BLEU	39.51	88.09	95.26	82.63	25.96	60.07
	XML-Validity	39.12	88.07	95.26	83.68	26.05	59.63
	XML-Match	39.39	88.15	95.26	87.37	29.56	60.36
	XML-BLEU	39.71	88.19	94.21	86.32	28.22	60.15
Ja→En	Prompt	44.14	85.93	90.53	80.00	22.38	65.25
	SFT	52.19	85.96	95.26	82.11	24.15	67.92
	FORMATRL *w/ Reward of:*						
	TreeSim	52.79	86.01	94.74	87.37	26.67	69.82
	Node-chrF	53.67	86.19	95.26	84.21	26.29	70.58
	Node-chrF (opt.)	53.20	86.03	94.74	84.74	25.96	69.70
	Content-BLEU	53.12	86.05	95.26	82.63	25.60	69.48
	XML-Validity	52.43	85.94	95.26	82.63	24.37	69.05
	XML-Match	53.12	85.98	95.26	86.32	26.45	69.26
	XML-BLEU	53.53	86.07	94.21	85.26	26.67	69.79

Table 4: Full evaluation on structured documents, contrasting Prompt, SFT, and FORMATRL with diverse reward functions.

Src→Tgt	Method	Translation		Structure		Combined	
		Content-BLEU	Content-COMET	XML-Validity	XML-Match	XML-BLEU	StrucAUC
En→Zh	Prompt	54.79	85.87	96.32	84.74	43.33	56.62
	SFT	57.95	86.19	98.42	89.47	47.51	63.14
	FORMATRL *w/ Reward of:*						
	TreeSim	57.70	86.22	98.42	90.00	47.63	64.29
	Node-chrF	57.80	86.39	97.89	89.47	47.64	64.46
	Node-chrF (opt.)	56.95	86.10	98.42	88.42	45.28	63.13
	Content-BLEU	58.08	86.23	97.89	88.95	47.00	63.01
	XML-Validity	57.15	86.22	98.42	88.95	46.50	63.93
	XML-Match	57.86	86.26	98.42	88.42	46.80	65.45
Zh→En	Prompt	39.92	83.31	95.26	83.68	33.83	67.06
	SFT	32.24	83.06	95.26	81.05	33.01	65.77
	FORMATRL *w/ Reward of:*						
	TreeSim	34.76	82.83	95.79	84.74	34.74	66.28
	Node-chrF	28.39	82.48	94.21	81.58	32.85	65.54
	Node-chrF (opt.)	29.71	82.60	94.21	80.53	32.50	65.11
	Content-BLEU	36.78	83.44	96.32	82.11	33.78	66.01
	XML-Validity	28.87	82.76	96.84	85.26	30.81	67.27
	XML-Match	35.87	82.65	94.74	80.53	32.03	65.94
	XML-BLEU	37.76	83.44	96.32	82.63	33.78	66.89
En→Ja	Prompt	40.13	87.90	96.32	79.47	26.78	45.07
	SFT	44.42	88.40	97.37	84.21	32.27	54.93
	FORMATRL *w/ Reward of:*						
	TreeSim	45.60	88.60	98.42	86.84	35.44	55.04
	Node-chrF	45.04	88.50	97.37	86.84	34.90	54.89
	Node-chrF (opt.)	44.11	87.87	97.89	85.26	34.40	53.12
	Content-BLEU	45.72	88.62	98.42	85.79	34.86	56.08
	XML-Validity	44.44	88.45	98.42	86.32	34.26	54.12
	XML-Match	44.14	88.40	97.37	86.32	34.22	54.11
	XML-BLEU	45.53	88.58	98.42	87.37	36.97	54.29
Ja→En	Prompt	35.61	84.86	98.95	81.58	27.58	64.65
	SFT	34.74	84.66	97.89	82.63	26.72	63.60
	FORMATRL *w/ Reward of:*						
	TreeSim	37.02	84.96	98.42	86.84	30.13	65.82
	Node-chrF	35.13	84.89	98.42	82.11	28.61	64.41
	Node-chrF (opt.)	32.61	84.91	98.95	83.16	28.98	64.46
	Content-BLEU	35.76	85.02	97.37	86.84	31.17	66.05
	XML-Validity	33.37	84.54	96.32	78.42	26.59	61.87
	XML-Match	33.50	84.72	97.89	82.63	27.66	64.09
	XML-BLEU	33.52	84.83	97.37	88.42	30.75	64.95

Table 5: Full evaluation on *inline markup* documents. Each language pair lists Prompt, SFT, and FORMATRL with different reward functions.



(a) enja (b) jaen

(c) enzh (d) zhen

Figure 10: Comparison of performance of different synthetic generation methods used in supervised fine-tuning, across four language pairs.

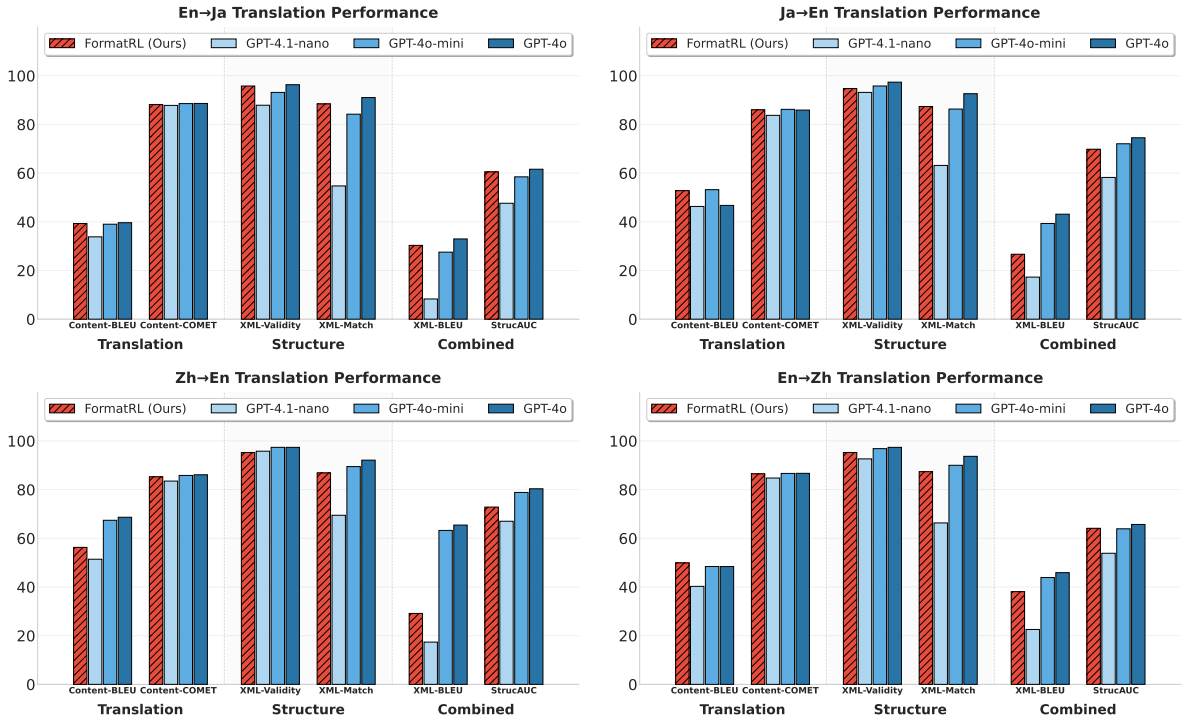


Figure 11: Comparison with GPT models across four language pairs.

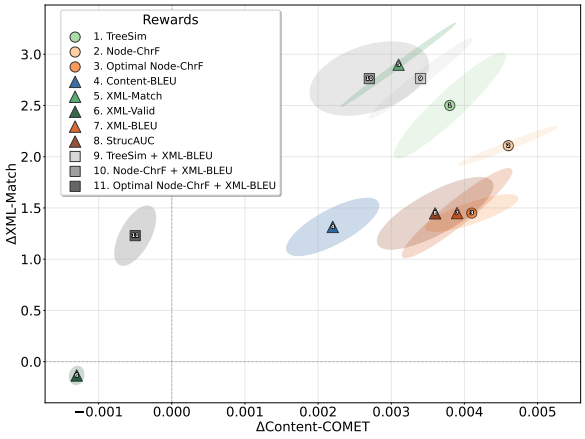


Figure 12: Improvement of FORMATRL over SFT using various single rewards, and combinations of two rewards. Points represent mean improvement and ellipses visualize the local covariance directional structure between two metrics improvements. Estimates are constructed from RL results based on 8 SFT checkpoints each.