FormatRL: Format Reinforcement Learning for Structured Document Translation

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@nict.go.jp

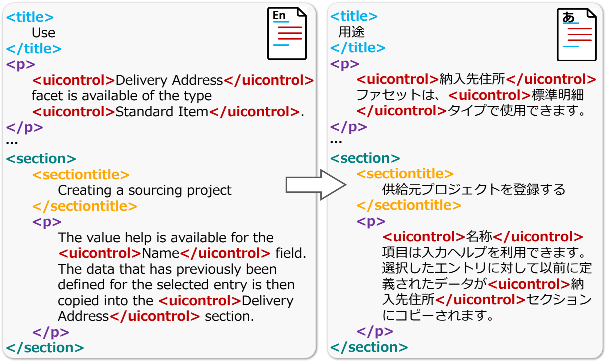
Abstract

Recent works on structured text translation remain limited to the sentence level, as they struggle to effectively handle complex document-level 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 reward, **TreeSim**, which measures structural similarity between predicted and reference XML trees. Experiments on the SAP software-documentation benchmark show improvements in both structural and translation quality over strong baselines.

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 [1, 2, 3]. 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 [4, 5]. Few-shot prompting is a convenient way to enable LLMs to learn markup transfer patterns with only a few examples [6, 7, 4], and fine-tuning provides more robust domain adaptation capabilities thus better performance [5]. 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.



```
<title>
    Use
</title>
<p>
    <uicontrol>Delivery Address</uicontrol>
    facet is available of the type
    <uicontrol>Standard Item</uicontrol>.
</p>
...
<section>
    <sectiontitle>
        Creating a sourcing project
    </sectiontitle>
    <p>
        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.
    </p>
</section>
```

```
<title>
  用途
</title>
<p>
    <uicontrol>納入先住所</uicontrol>
    ファセットは、<uicontrol>標準明細
    </uicontrol>タイプで使用できます。
</p>
...
<section>
    <sectiontitle>
        供給元プロジェクトを登録する
    </sectiontitle>
    <p>
        <uicontrol>名称</uicontrol>
        項目は入力ヘルプを利用できます。
        選択したエントリに対して以前に定
        義されたデータが<uicontrol>納
        入先住所</uicontrol>セクション
        にコピーされます。
    </p>
</section>
```

Figure 1: A structured document translation example (English→Japanese), with markup highlighted in color.

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 (GRPO) [8] with a novel structure-aware reward TreeSim for measuring XML tree structural similarity via edit distance from the XML tree of the reference document. The main contributions of this paper are summarized below:

- We propose **FormatRL** for structured document translation, with GRPO to optimize structural fidelity through a novel structure-aware reward **TreeSim**.
- Experimental results show significant improvements on SAP software documentation dataset, with FormatRL achieving average gains of 3.69 XML-Match and 0.25 Content-Bleu scores compared to the supervised fine-tuning baseline in four translation directions.

This paper is based on work published at IJCNLP-AACL 2025.

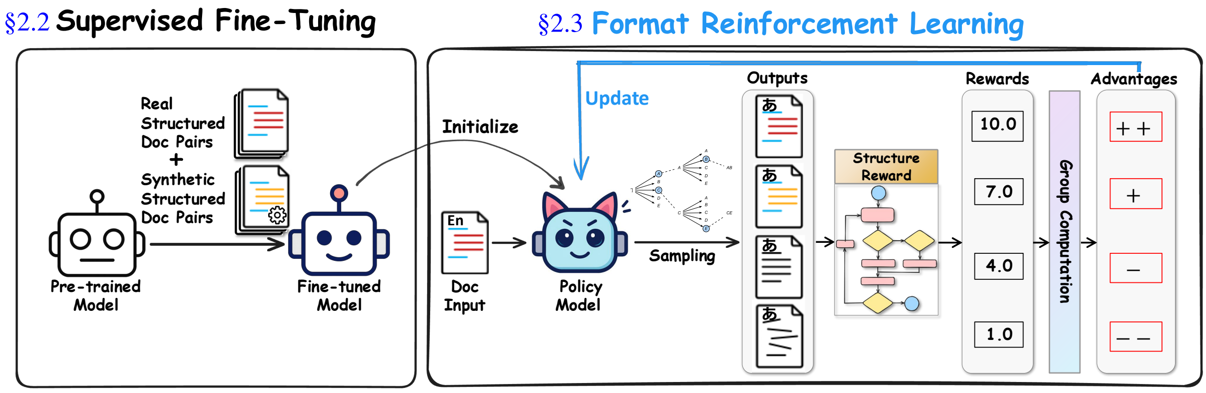


Figure 2: Our FormatRL pipeline consists of two stages: 1) fine-tune a LLM (e.g., Llama-3.1-8B-Instruct) using real and synthetic structured documents, and 2) reinforce the format handling ability using our proposed format reward TreeSim.

2 Method

Our pipeline is shown in Figure 2. We first define the task in § 2.1, then describe the supervised fine-tuning (SFT) phase in § 2.2, and finally present the core reinforcement learning phase in § 2.3.

2.1 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 $tag(v)$ (e.g., <p>) and may contain textual segments $text(v)$.

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

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

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 $\mathscr{D}_s$ and $\mathscr{D}_t$ 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:

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

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

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: 1) for any edge $(u, v) \in E_{D_s}$, we have $(\phi(u), \phi(v)) \in E_{\hat{D}_t}$, and 2) 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.

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 [9].

2.2 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. 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, and the original parallel texts are preserved. We ensure structural identity through validation and regeneration if fail until success or hitting a retry limit.

2.3 Phase II: Format Reinforcement

Initialized from the SFT checkpoint, we apply our designed reward to optimize the translation model (policy model as termed in GRPO) for structurally correct outputs.

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. We propose TreeSim to reinforce structure-aware similarity. **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 [10], 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. In practice, we scale each reward to $|r| \in [0, 10]$ for numerical stability. We also investigate the use of other metrics (§ 4) as rewards and explore combining two rewards by summing their scores.

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, resulting in the following objective:

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

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. And 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. In detail, $\hat{A}_i$ is computed as:

$$\begin{aligned}\hat{A}_i &= \frac{r(\hat{D}_{t,i}, D_t^\star) - \bar{r}}{\sigma_r} \\ \bar{r} &= \frac{1}{K}\sum_{j=1}^{K} r(\hat{D}_{t,j}, D_t^\star) \\ \sigma_r &= \sqrt{\frac{1}{K}\sum_{j=1}^{K}(r(\hat{D}_{t,j}, D_t^\star) - \bar{r})^2}\end{aligned}$$

3 Experimental Settings

3.1 Dataset

We use the SAP software documentation dataset [11] 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. Dataset statistics is shown in Appendix A.

3.2 Evaluation Metrics

We apply Content-Bleu and XML-Match as evaluation metrics. Content-Bleu measures translation quality using Bleu for a document with all XML markup removed, using the SacreBLEU [12] with language-specific tokenizers.[1] XML-Match measures structure correctness. It returns a binary score indicating whether the XML trees of output D_t and reference $D_t^\star$ are the same. Additionally, we report empirical p-values from statistical significance testing using bootstrap resampling with $1,000$ trials.

3.3 Implementation Details

We use Llama-3.1-8B-Instruct [13] as the base model. For the prompting baseline, we use in-context learning with 5 document pairs as exemplars. For supervised fine-tuning, we use GPT-4o to synthesize markup using the Asian Language Treebank (ALT) corpus [14, 15], generating 900 structured document pairs per language. The SFT model is trained on 100 real and 100 synthetic document pairs. For format reinforcement, we use K=8 generations per document with TreeSim as the reward function. See Appendix B for detailed hyperparameters.

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

4 Results and Analysis

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. First, FormatRL shows significant gains in structural preservation. XML-Match scores improve by an 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. 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.

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

Src→Tgt	Method	Content-BLEU	XML-Match
En→Zh	Prompt	49.88	76.84
	SFT	49.66	85.26
	FormatRL	**49.88**	**87.37**
Zh→En	Prompt	48.82	82.11
	SFT	**56.41**	83.68
	FormatRL	56.28	**86.84**
En→Ja	Prompt	36.60	67.37
	SFT	39.11	84.21
	FormatRL	**39.30**	**88.42**
Ja→En	Prompt	44.14	80.00
	SFT	52.19	82.11
	FormatRL	**52.79**	**87.37**

Comparison with Parse-and-Assemble. Parse-and-assemble baselines first extract translatable text blocks, then translate sentence-by-sentence, 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 3 shows that translation quality is comparable but FormatRL achieves higher XML-Match. We found parse-and-assemble methods struggle with in-line tags whose positions vary across target language syntax.

Reward Choice. Figure 4 shows the effect of different reward functions during GRPO training, including: 1) proposed TreeSim, 2) metrics used in evaluation as rewards, and 3) combination of two rewards. Estimates are constructed from the average of 4 translation directions.

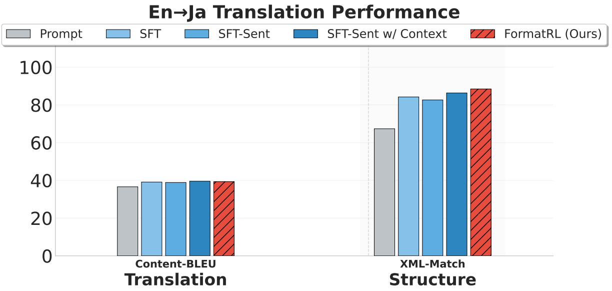


Figure 3: Comparison with parse-and-assemble baselines.

First, we found all rewards improve translation quality measured by Content-Bleu. Even pure structure-aware rewards, such as TreeSim and XML-Match, can improve translation. Second, we found 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 reward combination yields averaging effects, e.g., combining TreeSim with BLEU shows better Content-Bleu improvement than TreeSim alone.

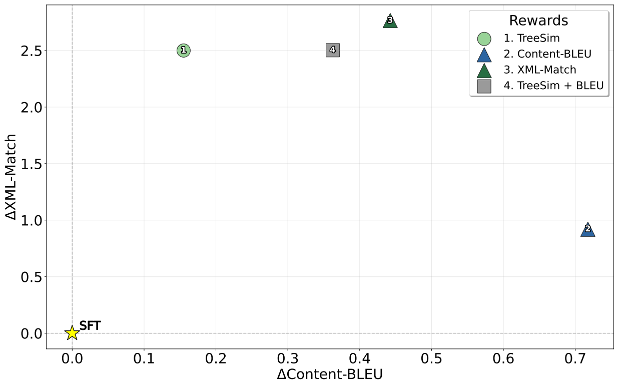


Figure 4: Improvement of FormatRL over SFT using various single rewards, and combinations of two rewards.

5 Conclusion

To address the challenge of translating documents with complex structures, we propose FormatRL, a novel reinforcement learning approach with proposed structure-aware reward TreeSim, which measures structural similarity between predicted and reference XML trees. Experimental results show FormatRL improves the structural fidelity of translated documents without compromising translation quality, compared with supervised fine-tuning and parse-and-assemble baselines.

Acknowledgements

This work was supported by JSPS KAKENHI Grant-in-Aid for Early-Career Scientists 25K21290.

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.

References

[1] Eric Joanis, Darlene Stewart, Samuel Larkin, and Roland Kuhn. Transferring markup tags in statistical machine translation: a two-stream approach. In Sharon O'Brien, Michel Simard, and Lucia Specia, editors, **Proceedings of the 2nd Workshop on Post-editing Technology and Practice**, Nice, France, September 2 2013.

[2] Mathias Müller. Treatment of markup in statistical machine translation. In Bonnie Webber, Andrei Popescu-Belis, and Jörg Tiedemann, editors, **Proceedings of the Third Workshop on Discourse in Machine Translation**, pp. 36–46, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[3] Thomas Zenkel, Joern Wuebker, and John DeNero. Automatic bilingual markup transfer. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, **Findings of the Association for Computational Linguistics: EMNLP 2021**, pp. 3524–3533, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

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

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

[6] 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, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, **Advances in Neural Information Processing Systems**, Vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020.

[7] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In **Advances in Neural Information Processing Systems**, Vol. 33, pp. 9459–9474, 2020.

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

[9] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In **Proceedings of the 40th annual meeting of the Association for Computational Linguistics**, pp. 311–318, 2002.

[10] Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. **SIAM J. Comput.**, Vol. 18, pp. 1245–1262, 12 1989.

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

[12] Matt Post. A call for clarity in reporting BLEU scores. In **Proceedings of the Third Conference on Machine Translation: Research Papers**, pp. 186–191, Belgium, Brussels, October 2018. Association for Computational Linguistics.

[13] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. The llama 3 herd of models. **arXiv**, 2024.

[14] Ye Kyaw Thu, Win Pa Pa, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. Introducing the Asian language treebank (ALT). In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, **Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)**, pp. 1574–1578, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

[15] 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. 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)**, pp. 1–6, 2016.

[16] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017.

A Dataset Statistics

Documents in the SAP 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.

B Hyperparameters

Supervised Fine-Tuning. We fine-tune for 20 epochs with batch size of 8, learning rate of 3×10^{-7}, cosine scheduling with warmup ratio 0.1, and AdamW optimizer [16]. Early stopping is triggered after 10 evaluations without improvement.

Format Reinforcement. We use learning rate 10^{-6}, train for 5 epochs, and set maximum sequence length to 2,000 tokens. The KL penalty β=0.01 and sampling temperature 1.0. We use per-device batch size 8 across 8 H200 GPUs (effective batch size 64). Early stopping is triggered after 3 evaluations without improvement.

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.