

STEER: Structured Text Editing via Edit-Chain Reinforcement Learning

Yucheng Lin Jiaojiao Jiang

University of New South Wales

High St, Kensington NSW 2052, Australia

{yucheng.lin, jiaojiao.jiang}@unsw.edu.au

Abstract

Text detoxification is increasingly used in content moderation and dialogue systems, where systems must produce safe rewrites while revealing what was changed and why. Existing approaches typically trade off interpretability and end-to-end optimization. Monolithic rewriting methods treat detoxification as a single opaque generation step, whereas edit-based pipelines make edits explicit but are trained in separate stages, preventing joint optimization. We propose STEER, a framework that models detoxification as a *structured edit chain* formulation and optimizes the entire editing process end-to-end with Group Relative Policy Optimization (GRPO). To make this formulation trainable, we construct synthetic edit-chain supervision using a multi-agent pipeline, perform supervised fine-tuning (SFT) to teach the model the structured editing format, and then apply GRPO with a veto-aligned multiplicative reward and granular format scoring. Experiments on ParaDetox and two out-of-domain benchmarks show that STEER achieves the highest aggregate J score among the compared systems on ParaDetox and remains competitive under out-of-domain evaluation.

Disclaimer: This paper describes toxic and discriminatory content that may be disturbing to some readers.

1 Introduction

As online platforms face increasing pressure to make content governance both effective and transparent, moderation systems are expected not only to handle harmful content but also to provide clear explanations for their decisions (European Parliament and Council of the European Union, 2022; European Commission, 2025). In this setting, text detoxification should no longer be viewed merely as an offline style-transfer benchmark (Logacheva et al., 2022; Dale et al., 2021; Khondaker et al., 2024). Practical systems must produce non-toxic

rewrites while also revealing what was changed and why.

Existing approaches to text detoxification largely fall into two paradigms. *Monolithic rewriting* methods (Logacheva et al., 2022; Dale et al., 2021; Khondaker et al., 2024) treat detoxification as a one-shot sequence transformation: the model directly rewrites a toxic input into a detoxified sentence in a single generation pass. This formulation enables end-to-end optimization over the final output, but all intermediate editing decisions remain implicit, making failures difficult to inspect or attribute. By contrast, *edit-based* methods (Li et al., 2018; Lee et al., 2024) decompose detoxification into explicit stages such as toxic-span detection, localization, deletion, and replacement. This makes the editing process more interpretable, but the stages are typically trained separately, which prevents global optimization over the entire editing process and often propagates errors across stages.

This tension suggests a natural goal for text detoxification: combining the inspectability of explicit editing with the optimization advantages of end-to-end generation. Recent reinforcement learning results make this combination increasingly plausible. In particular, GEM (Yu et al., 2025) shows that GRPO can substantially improve detoxification quality by directly optimizing generation with task-aligned reward signals. However, GEM still operates on monolithic rewriting over a flat text sequence. As a result, reinforcement learning improves the final rewrite, but it still cannot optimize or inspect individual editing decisions.

Motivated by this gap, we propose STEER (Structured Text Editing via Edit-chain Reinforcement Learning), a framework that represents detoxification as a structured edit chain and optimizes the chain end-to-end with GRPO. Instead of producing a detoxified sentence in one opaque step, STEER generates an explicit sequence of editing operations consisting of reasoning, span

localization, and replacement. This representation preserves a transparent record of the model’s decisions while allowing reinforcement learning to optimize the entire editing process as a single trajectory.

We instantiate this formulation with a three-stage pipeline: multi-agent construction of synthetic edit chains, LoRA-based SFT for format acquisition, and GRPO for end-to-end optimization of full edit trajectories. This design addresses the absence of edit-chain annotations, the lack of native structured-output ability in pretrained LLMs, and the format instability of early-stage RL.

Experiments on ParaDetox and two out-of-domain benchmarks show that STEER achieves the highest aggregate J score among the compared systems on ParaDetox and remains competitive under out-of-domain evaluation. Further analysis shows that STEER concentrates edits on toxic spans rather than rewriting entire sentences, making model behavior easier to inspect and diagnose.

Our contributions are summarized as follows:

- **Structured optimization with inspectability.** We formulate detoxification as a structured edit chain that exposes individual editing decisions while supporting end-to-end optimization.
- **A trainable edit-chain pipeline.** We construct synthetic edit-chain supervision, teach the format with SFT, and stabilize structured-output RL with granular format scoring.
- **Evidence for the quality–transparency trade-off.** Across in-domain and out-of-domain benchmarks, STEER improves joint detoxification quality while producing more localized, diagnosable edits.

2 Related Work

2.1 Monolithic Text Detoxification

Most text detoxification methods formulate the task as monolithic rewriting, where a model directly generates a detoxified sentence in a single step. Representative examples include supervised sequence-to-sequence training on ParaDetox (Lo-gacheva et al., 2022), discriminator-guided generation such as ParaGeDi (Dale et al., 2021), LLM-based pseudo-parallel training in DetoxLLM (Khondaker et al., 2024), prompt tuning in YOPO (He et al., 2023), and GRPO-based optimization

in GEM (Yu et al., 2025). Despite differences in supervision and scale, these methods optimize only the final rewrite and leave intermediate editing decisions latent.

2.2 Edit-Based and Interpretable Rewriting

A separate line of work exposes the rewriting process through explicit editing signals. Delete-Retrieve-Generate (Li et al., 2018) decomposes style transfer into deletion, retrieval, and insertion, while XDetoX (Lee et al., 2024) uses token-level toxicity explanations to guide masking and infilling. Reasoning-style supervision has also been explored in CoTeX (Zhang et al., 2024) and Detox-CoT (Tang et al., 2024). We build on this direction but optimize the full edit trajectory rather than a staged pipeline.

We draw inspiration from adjacent text editing tasks, where models such as the Levenshtein Transformer (Gu et al., 2019) and GECToR (Omelianchuk et al., 2020) show that text generation can be formulated as explicit sequences of edits rather than one-shot rewriting. Our work brings this perspective into the field of text detoxification and further combines it with reinforcement learning over structured edit trajectories.

2.3 Reinforcement Learning for Structured Generation

Reinforcement learning is increasingly used to align language models with structured objectives. PPO-style RLHF (Ouyang et al., 2022; Schulman et al., 2017) and GRPO (Shao et al., 2024) optimize generation from external rewards, while later work shows that format-aware rewards can encourage structured reasoning and improve stability (Guo et al., 2025; Liu et al., 2025). In detoxification, GEM (Yu et al., 2025) applies GRPO to monolithic rewriting; in contrast, we apply GRPO to structured edit trajectories.

3 Method

In this section, we introduce our text detoxification framework, STEER. As shown in Figure 2, STEER proceeds in three stages. First, a multi-agent pipeline is formulated to construct synthetic edit-chain supervision from parallel detoxification data (§3.2). Then, a LoRA-based SFT is developed to teach the model to generate valid, executable edit chains in the required format (§3.3). Finally, a GRPO-based reinforcement learning is trained to

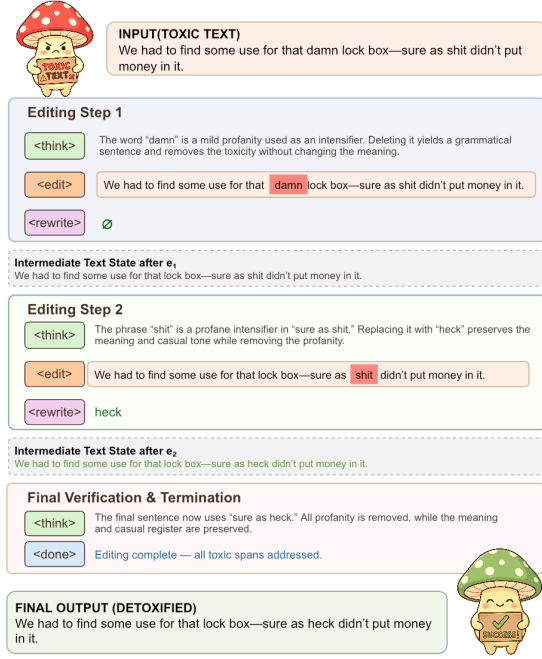


Figure 1: A source-backed two-step edit chain for ParaDetox test instance 594 (zero-based). Text is detoxified and reasoning is abridged for readability; all edits and termination match the model output.

optimize the edit trajectories to ensure detoxification quality, semantic preservation, and structural validity (§3.4).

Before we introduce the three stages in STEER, we first explain the formulation of edit chain used to model detoxification trajectories (§3.1).

3.1 Edit Chain Formulation

Given a toxic input text x , the structured edit chain is formulated as

$$c = (e_1, e_2, \dots, e_K, \langle \text{done} \rangle),$$

where K denotes the number of edit steps and each edit step in c is defined as

$$e_k = (t_k, l_k, r_k).$$

Here, t_k denotes a reasoning segment, l_k denotes the source span selected for editing, and r_k denotes the replacement text. In the model format, these components are realized through the structured tags `<think>`, `<edit>`, and `<rewrite>`, respectively. After all edits have been generated, the model emits the termination token `<done>`.

A non-toxic rewrite y is obtained by executing the edit steps e_k sequentially. Figure 1 presents an example edit chain with two edit steps. As we can

see that the edit chain preserves an explicit record of the editing process while allowing the entire edit trajectory to be generated by a single model.

3.2 Structured Data Generation

Existing detoxification datasets typically provide only parallel sentence pairs (x, y^*) , where x is a toxic input sentence and y^* is its human-written detoxified rewrite. However, they do not annotate the intermediate editing process that transforms x into y^* . In this subsection, we introduce a two-agent pipeline that reconstructs the intermediate editing process from (x, y^*) . The two agents operate under strict information asymmetry, enabling the step-by-step reasoning, span localization, and replacement decisions that form the edit chain.

Editor Agent. The Editor receives the toxic input x together with an instruction specifying the edit-chain output formulation, including the required `<think>`, `<edit>`, `<rewrite>`, and `<done>` fields. It then generates a candidate edit chain c without access to the human rewrite y^* . This information restriction prevents direct copying of the reference rewrite y^* and forces the model to learn more generalizable editing patterns.

Reviewer Agent. The Reviewer receives the toxic input x , the reference rewrite y^* , and the candidate edit chain c , and verifies c with three tool-assisted checks. First, a *Format Validator* checks whether c is structurally valid, i.e., whether it follows the required edit-chain schema with correctly ordered `<think>`, `<edit>`, and `<rewrite>` fields, terminates with `<done>`, and can be deterministically parsed and executed to produce a rewritten output $\hat{y}$. If this check fails, the candidate edit chain c is immediately rejected and returned to the Editor with structured feedback, without running the later checks. Second, an *Auto Scorer* computes the cosine similarity between x and $\hat{y}$, and accepts the candidate only if the similarity exceeds 0.6. Third, a *Leak Detector* checks for reference leakage based on n -gram overlap between the reasoning segments in c and the human rewrite y^* . Only candidates that pass all three checks are accepted as supervision; otherwise, they are returned to the Editor for another attempt, while the reference rewrite itself is never revealed.

3.3 Supervised Fine-Tuning (SFT)

Using the structured edit chains constructed in §3.2, we now perform SFT to train the model to generate

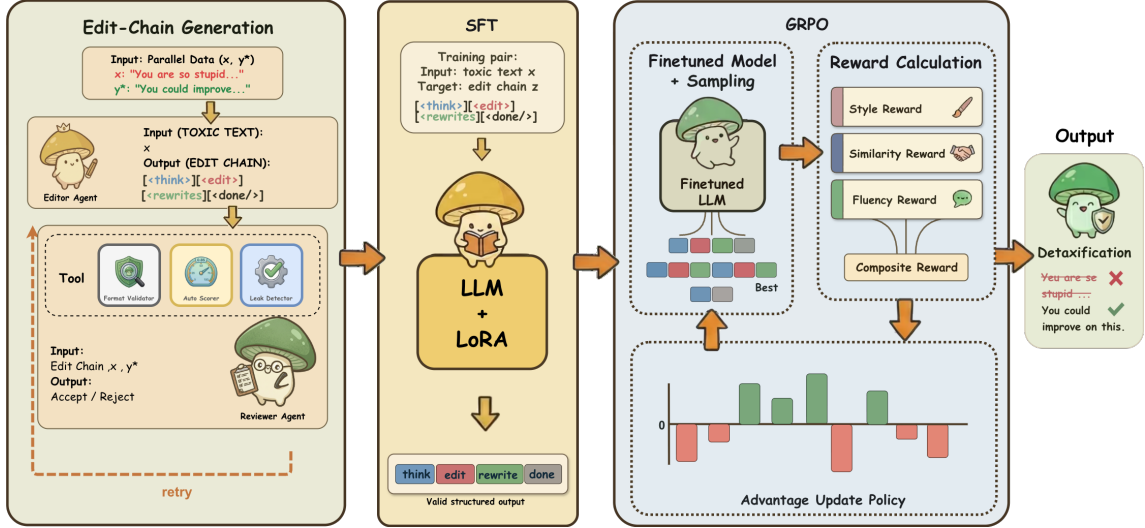


Figure 2: Overview of STEER. A multi-agent pipeline constructs edit chains from parallel pairs (x, y^*) . Stage 1 performs LoRA SFT to learn the structured `<think>/<edit>/<rewrite>/<done>` format. Stage 2 applies GRPO, using Eq. (3) for valid outputs and granular format scoring (Eq. 4) for malformed ones.

valid edit trajectories in this subsection. This stage provides a cold-start initialization that teaches the model to follow the edit-chain format, forming the basis for subsequent reinforcement learning.

Let (x_i, z_i) denote the i -th training example, where x_i is a toxic input text and z_i is the corresponding edit-chain sequence generated by the data construction pipeline. We train the model using a standard cross-entropy objective

$$\mathcal{L}_{\text{SFT}} = -\frac{1}{N} \sum_{i=1}^N \log P_{\theta}(z_i | x_i), \quad (1)$$

where N is the number of training examples, x_i is the i -th toxic input, z_i is the corresponding edit chain sequence, and $P_{\theta}(\cdot | x_i)$ is the conditional distribution defined by model parameters θ . To reduce the number of trainable parameters and stabilize training, we adopt LoRA (Hu et al., 2021) for parameter-efficient fine-tuning.

3.4 GRPO Optimization

After SFT establishes basic adherence to the required edit chain format, we further optimize edit-chain generation with GRPO in this subsection.

An output edit chain is considered *structurally valid* if it follows the required edit-chain formulation, contains correctly ordered `<think>`, `<edit>`, and `<rewrite>` fields, terminates with `<done>`, and can be deterministically parsed and executed. Outputs that violate any of these conditions are treated as *malformed*.

3.4.1 Reward Design

A suitable reward for detoxification should reflect its conjunctive objective: high-quality outputs must jointly satisfy non-toxicity, semantic preservation, and fluency, rather than performing well on only a subset of these dimensions. We therefore adopt a multiplicative reward, so that strong performance on one dimension cannot offset severe deficiencies in another:

$$R_{\text{raw}} = \text{STA}^{\alpha} \times \text{SIM}^{\beta} \times \text{FL}_{\text{soft}}^{\gamma}, \quad (2)$$

where STA denotes the style transfer accuracy score, measuring the extent to which the rewritten sentence is judged non-toxic; SIM denotes semantic similarity between the input and output sentences; and $\text{FL}_{\text{soft}} = \max(\text{FL}_{\text{prob}}, \tau_f)$, where FL_{prob} denotes the fluency score predicted by a grammatical acceptability model, and τ_f is a lower bound on the fluency term. This prevents extremely low fluency scores from prematurely collapsing the reward product and preserves informative gradient signal from the remaining dimensions. The exponents α , β , and γ reshape the relative sensitivity of the three reward components.

To improve optimization stability, we further shift, scale, and clip the raw reward R_{raw} :

$$R = \text{clip}((R_{\text{raw}} - b) \cdot s, r_{\min}, r_{\max}), \quad (3)$$

where b is a baseline shift, s is a scaling factor, and $r_{\min}$ and $r_{\max}$ are clipping bounds.

If every malformed output receives the same flat penalty, then whenever all responses within

a sampled group are invalid, the within-group reward variance collapses to zero. This causes all advantages to become zero, thereby eliminating the learning signal. To mitigate this failure mode, we assign each malformed output o a granular format score $R_{\text{fmt}}(o)$ instead of a flat penalty. Starting from a base penalty r_{base} , we add a fixed bonus δ for each correctly formed structural element among `<think>`, `<edit>`, `<rewrite>`, and `<done>`, and cap the result with a hard ceiling r_{ceil} . The granular format score is defined as

$$R_{\text{fmt}}(o) = \min \left(r_{\text{base}} + \delta \sum_{j=1}^4 \mathbb{I}[e_j \in o], r_{\text{ceil}} \right), \quad (4)$$

where $\mathbb{I}[e_j \in o]$ equals 1 if the j -th structural element is present in o and 0 otherwise. The hard ceiling satisfies $r_{\text{ceil}} < r_{\text{min}}$, which guarantees that the reward ordering is never inverted, so every valid edit chain scores higher than every malformed output, while malformed outputs of different quality still induce non-zero within-group variance.

3.4.2 GRPO Objective

For each toxic input text x , we sample a group of G candidate edit-chain outputs from the frozen old policy $\pi_{\theta_{\text{old}}}$, denoted by $\{o_1, \dots, o_G\}$. Structurally valid outputs are scored with the task reward in Eq. (3), whereas malformed outputs receive the granular format score in Eq. (4). Following Dr. GRPO (Liu et al., 2025), we convert these scores into mean-centered within-group advantages:

$$\hat{A}_j = R(o_j) - \frac{1}{G} \sum_{k=1}^G R(o_k), \quad (5)$$

where $\hat{A}_j$ denotes the advantage of the j -th sampled output o_j relative to the mean reward of its group; no standard-deviation normalization is applied.

We optimize a dual-clipped PPO-style surrogate with KL regularization toward the SFT-initialized reference policy:

$$\begin{aligned} q_{j,t} &= \min \left(\rho_{j,t} \hat{A}_j, \text{clip}(\rho_{j,t}, 1-\epsilon, 1+\epsilon) \hat{A}_j \right), \\ \mathcal{S}_{j,t} &= \begin{cases} q_{j,t}, & \hat{A}_j \geq 0, \\ \max(q_{j,t}, c\hat{A}_j), & \hat{A}_j < 0, \end{cases} \\ \mathcal{J}_{\text{GRPO}}(\theta) &= \mathbb{E} \left[\sum_{j=1}^G \sum_t \mathcal{S}_{j,t} \right] - \lambda_{\text{KL}} D_{\text{KL}}(\pi_{\theta} \parallel \pi_{\text{ref}}). \end{aligned} \quad (6)$$

Here $\rho_{j,t} = \pi_{\theta}(w_{j,t} \mid x, w_{j,<t}) / \pi_{\theta_{\text{old}}}(w_{j,t} \mid x, w_{j,<t})$ is the token-level importance ratio between the current policy and the frozen old policy, ϵ is the PPO clipping coefficient, and $c=3.0$ is the dual-clip coefficient for negative advantages. The fixed reference policy π_{ref} is used only for KL regularization, controlled by λ_{KL} .

STEER optimizes detoxification quality, semantic preservation, structural validity, and policy stability at the level of the full edit trajectory, thereby combining explicit editing decisions with end-to-end reinforcement learning in a single framework.

4 Experiments

In this section, we first introduce the experimental setting. Then, we evaluate STEER in both in-domain and out-of-domain settings against a broad set of strong baselines. We further conduct ablation study. Finally, we conduct a human evaluation on sampled ParaDetox test instances.

4.1 Experimental Setup

Datasets. We evaluate on ParaDetox (Logacheva et al., 2022), a human-annotated parallel detoxification dataset with $\sim 11\text{K}$ sentence pairs, using the standard test split of 671 examples. For out-of-domain evaluation, we test the ParaDetox-trained models without adaptation on a 383-example filtered test subset of DetoxLLM (Khondaker et al., 2024) and a Hugging Face detoxification dataset.

Baselines. We compare against representative supervised, edit-based, prompting, and LLM fine-tuning baselines, including ParaDetox, BART/T5 SFT, COUNT, XDetoX, DiffuDetox, DetoxLLM, YOPO, 5-shot Llama/Qwen, monolithic Llama/Qwen SFT, and GEM. The most direct comparisons are monolithic SFT and GEM, which use the same backbone family as ours but operate without structured edit chains. For methods that release only checkpoints, we evaluate the released models directly. Baseline results marked with superscripts are taken from the corresponding original papers.

Metrics. Following Logacheva et al. (2022) and Yu et al. (2025), we report style transfer accuracy (STA), semantic similarity (SIM), fluency (FL), and the aggregate metric J . STA is computed with a Jigsaw-trained toxicity classifier, SIM with Wieting word-averaging embeddings (Wieting et al., 2019), and FL with a CoLA-trained RoBERTa

Table 1: In-domain results on ParaDetox. **Bold**: best within each group. †: evaluated on the ParaDetox test set without training on ParaDetox.

Method	STA	SIM	FL	J†
Human reference	95.53	77.33	88.23	65.36
<i>Traditional Supervised</i>				
ParaDetox ^a	90.31	85.77	88.97	67.83
BART SFT ^a	87.18	86.68	89.12	66.40
T5 SFT ^a	62.74	88.64	88.25	46.89
<i>Edit-based Pipelines</i>				
COUNT ^b	90.46	85.03	87.28	67.39
XDetoX ^c	93.74	84.23	86.29	68.15
DiffuDetox ^d	95.68	77.49	88.67	66.00
<i>Prompt / ICL</i>				
DetoxLLM ^{e†}	95.38	59.15	97.62	54.70
YOPO ^f	82.27	89.40	87.03	62.56
Llama 5-shot	66.32	58.58	96.72	37.49
Qwen 5-shot	89.12	75.23	96.42	63.91
<i>LLM Fine-tuning</i>				
Llama SFT	86.14	83.12	93.14	65.36
Qwen SFT	90.31	83.05	90.16	66.94
STEER-Llama (SFT)	88.23	88.66	90.61	70.30
STEER-Qwen (SFT)	91.06	87.88	90.01	72.53
<i>SFT + RL</i>				
GEM (Llama) ^g	95.98	82.39	88.38	69.61
GEM (Qwen) ^g	93.74	83.93	87.33	68.26
STEER-Llama (GRPO)	94.93	89.30	91.36	77.69
STEER-Qwen (GRPO)	94.63	88.42	90.91	76.25

^aLogacheva et al. (2022) ^bPour et al. (2023) ^cLee et al. (2024) ^dFloto et al. (2023) ^eKhondaker et al. (2024) ^fHe et al. (2023) ^gYu et al. (2025)

model (Warstadt et al., 2019). The aggregate metric J is defined as follows

$$J = \frac{1}{N} \sum_{i=1}^N \text{STA}_i^{\text{bin}} \cdot \text{SIM}_i \cdot \text{FL}_i^{\text{bin}}, \quad (7)$$

where N is the number of test samples, and $\text{STA}_i^{\text{bin}}$ and FL_i^{bin} denote the binarized style and fluency judgments in the benchmark protocol of Yu et al. (2025).

Implementation. We use Llama-3.1-8B-Instruct¹ and Qwen-2.5-7B-Instruct² as backbones. Both follow the same pipeline: synthetic edit-chain construction, LoRA SFT, and GRPO. SFT is trained for 3 epochs on 9,095 annotated samples; GRPO uses learning rate 3×10^{-6} , group size 16, temperature 0.70, and KL coefficient 0.02. Full training hyperparameters are reported in Appendix B.

¹<https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct>

²<https://huggingface.co/Qwen/Qwen2.5-7B-Instruct>

To reduce proxy overlap between training and evaluation, we use reward models that differ from the benchmarking scorers on the two most discriminative dimensions. STA is computed with a RoBERTa model trained on TweetEval (Babieri et al., 2020), and SIM is computed with all-mpnet-base-v2 (Reimers and Gurevych, 2019). FL is computed with a CoLA-trained RoBERTa model (Warstadt et al., 2019), which shares the same model family as the evaluation scorer. However, as shown in Table 1, FL is the least discriminative dimension in practice, with all competitive methods scoring above 87%, so this residual overlap has limited impact on the relative ranking of methods. For all STEER and monolithic SFT/GRPO variants, we train three independent runs with different random seeds and report the mean across runs.

4.2 In-Domain Evaluation

Table 1 shows the in-domain evaluation results. STEER achieves the strongest overall performance on ParaDetox. STEER-Llama (GRPO) obtains the best J score at 77.7 (77.69 before rounding), followed by STEER-Qwen (GRPO) at 76.25.

Compared with monolithic SFT using the same backbone, edit-chain SFT already brings a substantial gain. For Llama, J increases from 65.36 to 70.30, with the largest improvement coming from semantic preservation as SIM rises from 83.12 to 88.66. Applying GRPO on top of this further improves J from 70.30 to 77.7, mainly through a large increase in STA from 88.23 to 94.93 while largely retaining the semantic advantage.

The comparison with GEM is particularly informative because both methods use GRPO while differing in output representation. With the same Llama backbone, STEER-Llama (GRPO) improves over the reported GEM (Llama) result by 8.08 J points and also achieves higher SIM and FL. This benchmark-level comparison is consistent with a benefit from structured edit trajectories, although differences in training and reward settings prevent us from attributing the gain solely to the representation. Paired-bootstrap tests on per-example J further show that GRPO significantly improves over edit-chain SFT across both backbones and all three test sets (all $p \leq 0.0006$); detailed confidence intervals are reported in Appendix F.

STEER also obtains a higher automatic J score than the human reference, likely because the benchmark favors local, minimal edits whereas human

Table 2: Out-of-domain results. **Bold**: best within each group. *DetoxLLM is evaluated on its own corpus. All STEER variants use the ParaDetox-trained checkpoint without adaptation.

Method	DetoxLLM				HuggingFace			
	STA	SIM	FL	J↑	STA	SIM	FL	J↑
Original ref.	97.1	44.9	98.2	43.0	60.2	74.0	89.4	40.0
<i>Traditional Supervised</i>								
ParaDetox ^a	59.3	84.2	87.2	41.4	78.0	91.9	93.2	65.5
BART SFT ^a	59.5	81.8	89.6	39.6	77.4	89.9	94.4	64.6
T5 SFT ^a	36.8	86.5	89.9	27.3	65.4	91.4	94.2	55.1
<i>Edit-based Pipelines</i>								
COUNT ^b	63.2	78.4	89.0	42.0	80.6	89.5	91.6	65.4
XDetoX ^c	89.3	79.3	91.4	63.4	98.0	77.4	93.0	70.8
DiffuDetox ^d	79.4	66.4	29.8	10.7	60.0	74.1	90.0	40.6
<i>Prompt / ICL</i>								
DetoxLLM ^{e*}	95.0	49.4	98.4	46.1	94.2	61.0	98.6	55.6
YOPO ^f	51.4	91.1	88.5	39.3	72.8	95.2	92.8	62.9
<i>LLM Fine-tuning</i>								
Llama SFT	64.5	78.5	91.4	44.5	86.4	83.9	96.4	68.7
Qwen SFT	64.5	76.5	87.0	40.6	83.2	86.0	94.0	65.6
STEER-Llama (SFT)	65.3	86.6	85.6	47.8	88.6	86.9	90.0	68.7
STEER-Qwen (SFT)	67.4	85.4	86.7	50.5	87.4	85.1	91.0	66.8
<i>SFT + RL</i>								
GEM (Llama) ^g	71.0	83.3	88.3	51.3	89.0	85.4	96.2	72.7
GEM (Qwen) ^g	73.9	79.6	90.3	51.9	91.2	84.1	94.0	71.4
STEER-Llama (GRPO)	79.6	87.2	87.7	60.5	93.8	86.6	91.4	74.0
STEER-Qwen (GRPO)	78.3	86.4	88.0	59.4	91.8	86.4	90.8	71.0

^aLogacheva et al. (2022) ^bPour et al. (2023) ^cLee et al. (2024) ^dFloto et al. (2023) ^eKhondaker et al. (2024) ^fHe et al. (2023) ^gYu et al. (2025)

references often include broader stylistic revisions. We therefore treat the human reference as a point of comparison rather than as a strict upper bound under this metric. Additional in-domain analysis of editing operations and interpretability on ParaDetox is provided in Appendix C.

4.3 Out-of-Domain Generalization

To evaluate the out-of-domain generalization of STEER, we apply the ParaDetox-trained models to a 383-example filtered DetoxLLM test subset³ and the Hugging Face detoxification dataset⁴ without domain adaptation. The DetoxLLM row reports its original in-domain setting, whereas all other methods, including STEER, are evaluated using ParaDetox-trained checkpoints without adaptation.

Table 2 shows the out-of-domain results of different methods. STEER remains strongly competitive under distribution shift. On HuggingFace, STEER-Llama (GRPO) achieves the best overall result at 74.0 J. On DetoxLLM, it reaches 60.5 J, trailing

only XDetoX at 63.4 J while outperforming all other LLM-based baselines. Both STEER variants also exceed DetoxLLM despite being evaluated fully cross-domain.

Compared with monolithic Llama SFT, STEER-Llama (GRPO) improves J by 16.0 points on DetoxLLM and by 5.3 points on HuggingFace. Compared with GEM (Llama), it improves J by 9.2 and 1.3 points on the two benchmarks. This suggests that the edit-chain representation transfers beyond the ParaDetox training distribution.

On DetoxLLM, XDetoX achieves higher J mainly through stronger STA, whereas STEER retains a larger semantic advantage. We further analyze hard lexicon-unmatched and stance-bearing subsets in Appendix D, showing that STEER’s gains are not reducible to dictionary-triggered deletion or replacement.

4.4 Ablation Study

To isolate the contribution of each design choice, we use Llama-3.1-8B-Instruct as the backbone and progressively add one component at a time.

The ablation study results are shown in Table 3. As we can see that each component contributes

³<https://github.com/allacnobuf/Detoxification-of-Text/blob/main/data/detoxtest.json>

⁴https://huggingface.co/datasets/narySt/text_detoxification_dataset

Table 3: Ablation study on ParaDetox (Llama backbone). Each row adds one component to the previous. ΔJ shows the marginal improvement. $\dagger$ Standard SFT uses Llama-3.1-8B full-data SFT from Yu et al. (2025).

Configuration	STA	SIM	FL	J $\uparrow$	ΔJ
Standard SFT $\dagger$	86.1	83.1	93.1	65.4	–
Edit chain SFT	88.2	88.7	90.6	70.3	+4.9
+ GRPO (additive)	95.4	88.0	89.4	75.4	+5.1
+ Granular format	95.4	88.1	90.8	76.3	+0.9
+ STEER	94.9	89.3	91.4	77.7	+1.4

to the final performance and that their effects are complementary. Replacing standard SFT with edit-chain SFT yields a gain of 4.9 J points, driven mainly by a large increase in SIM. This supports the view that explicit localization and local editing reduce unnecessary rewriting. Adding GRPO on top of edit-chain SFT gives the largest single improvement, 5.1 J points, mainly through a substantial increase in detoxification success. Granular format scoring and the multiplicative reward each provide smaller but consistent gains, improving training stability and objective alignment. Overall, STEER improves over standard SFT by 12.3 J points. We also explored several alternative designs for token-level credit assignment and reward shaping, but did not observe consistent gains in our setting; these negative results are summarized in Appendix H.

4.5 Human Evaluation

To complement automatic metrics, we conduct a human evaluation on 200 randomly sampled ParaDetox test instances, comparing STEER (GRPO), GEM, and STEER (SFT). The annotation was performed by three annotators from different nationalities, all with graduate-level NLP research experience. The annotation protocol proceeds in two strictly isolated rounds. In Round 1 (blind evaluation), annotators see only the source text and three anonymized system outputs (in randomized order) and rate each output on detoxification (DET), semantic preservation (SEM), and fluency (FLU) using a 1–5 Likert scale ranging from “completely fails” to “fully succeeds.”⁵ A content warning about toxic language was provided beforehand. In Round 2, annotators are shown the full edit chain produced by STEER and evaluate the structural

⁵DET measures whether toxic content is removed, SEM measures whether the original meaning is preserved, and FLU measures whether the output is grammatically correct and natural.

Table 4: Human Likert evaluation on 200 ParaDetox samples (1–5 scale). Inter-annotator agreement is measured by quadratic-weighted Cohen’s κ (Cohen, 1968), averaged over annotator pairs. Best in **bold**.

System	DET $\uparrow$	SEM $\uparrow$	FLU $\uparrow$
STEER (GRPO)	4.45 $\pm$.77	4.65$\pm$.74	4.42 $\pm$.77
GEM	4.58$\pm$.71	3.53 $\pm$ 1.02	4.55$\pm$.55
STEER (SFT)	4.22 $\pm$ 1.15	4.78 $\pm$.57	4.28 $\pm$.95
κ_w (QWK)	0.430	0.727	0.281

consistency discussed in the operation-level analysis above. Annotators cannot retroactively modify Round 1 scores, preventing the reasoning chain from influencing quality judgments.

Table 4 summarizes the Likert results, and Appendix G reports descriptive pairwise comparisons derived from the same ratings. The main finding is a large semantic advantage: STEER scores 1.13 points higher than GEM on SEM ($p < 0.001$), while GEM leads narrowly on DET and FLU (+0.13 each). Given the relatively low inter-annotator agreement for FLU ($\kappa_w = 0.281$), the small fluency difference should be interpreted cautiously. Aggregating across dimensions, STEER (GRPO) averages 4.51 versus GEM’s 4.22. The higher derived score of STEER (SFT) over GEM is consistent with a benefit from structured edit-chain supervision, but it does not isolate the representation under fully controlled training conditions.

5 Conclusion

In this paper, we identified a central limitation of existing text detoxification methods: they typically favor either end-to-end optimization or interpretable editing, but rarely achieve both at once. To address this gap, we proposed STEER, a framework that models detoxification as a structured edit chain and optimizes the full editing process end-to-end with GRPO. To make this feasible, we constructed synthetic edit-chain supervision, used SFT to teach the structured format, and introduced reward designs that stabilize RL over structured outputs.

Experimental results show that STEER achieves state-of-the-art performance on ParaDetox and remains strongly competitive under domain shift, consistently outperforming strong monolithic baselines in semantic preservation and overall joint quality. Further analysis shows that STEER tends to make localized, inspectable edits concentrated on toxic spans.

Limitations

Despite the strong results, our approach still has several limitations. (1) **Generalization beyond sentence-level detoxification remains limited.** Our experiments focus on sentence-level rewriting, where toxic content is often localized in short spans. Although STEER shows strong transfer to two out-of-domain benchmarks, we have not yet validated it on document-level editing, multi-sentence discourse settings, or more complex rewriting tasks such as multi-attribute transfer. These settings may require longer-horizon edit planning and cross-sentence consistency that are not captured by the current formulation.

(2) **Inference efficiency is weaker than that of direct rewriting methods.** Because STEER generates explicit reasoning, span localization, and replacement steps, its outputs are longer than those of monolithic detoxification models, which increases decoding cost and inference latency. Under matched batched vLLM inference on the 671 ParaDetox samples with the same Llama backbone, STEER takes 8.1 ms/sample compared with GEM’s 5.6 ms/sample, a $1.45\times$ throughput cost for an 8.08-point gain in J and an auditable edit trace. Distillation, step compression, or adaptive early stopping may help mitigate this issue.

(3) **The learned policy is still influenced by imperfect reward and evaluation proxies.** Although we deliberately decouple the reward scorers from the benchmark evaluation scorers for STA and SIM, each individual classifier still carries its own inductive biases, which may propagate into policy optimization. In particular, the fluency dimension still relies on the same CoLA-based model family in both training and evaluation. Moreover, the generated `<think>` traces provide an interpretable record of the editing process, but they should not be taken as fully faithful explanations of the model’s internal decision mechanism.

(4) **The data construction pipeline currently depends on synthetic supervision from a strong external model.** Our edit-chain annotations are produced through a multi-agent pipeline whose Editor backbone is GPT-4o. GPT-4o is used only for cold-start edit-chain construction; the final STEER models use open-weight Llama and Qwen backbones, and inference does not require any external API. Still, this design introduces dependence on external model quality and API accessibility, and the synthetic edit chains may not fully reflect the di-

versity of human editing strategies. Replacing the teacher with stronger open-source models and incorporating more human-validated edit trajectories would be important directions for future work.

Ethics Statement

This work studies text detoxification using public datasets that contain toxic and hateful language (Logacheva et al., 2022; Khondaker et al., 2024); a content warning is provided at the beginning of the paper. Toxicity classifiers may mis-handle identity terms, dialectal expressions, reclaimed slurs, implicit toxicity, and politically or culturally contextualized language. STEER may therefore over-detoxify legitimate expression, erase minority language varieties, or alter a speaker’s intended stance. We intend STEER only as an assistive rewriting component, not an autonomous moderation policy. Deployments should preserve the original text, expose proposed edits, provide human oversight, and allow users to reject or undo a rewrite.

The human evaluation involved three graduate-level NLP annotators from different institutions and nationalities. They were warned in advance about toxic content, participated voluntarily, could stop at any time, and were compensated above the applicable local minimum wage. Because the editing direction could in principle be reversed to produce toxic content, access and deployment should be monitored; the explicit edit-chain format nevertheless makes proposed changes easier to inspect than opaque rewriting.

Reproducibility

The data construction procedure and filtering criteria are described in §3.2 and Appendix A. Training and evaluation settings are reported in §4.1 and Appendix B, with sensitivity analyses in Appendix E and paired-bootstrap details in Appendix F.

Acknowledgments

This work was supported by the Australian Research Council (ARC) under the DECRA Fellowship DE240101049. This research includes computations using the computational cluster Katana, supported by Research Technology Services at UNSW Sydney (PVC (Research Infrastructure), UNSW Sydney, 2010).

References

- Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. Tweeteval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the association for computational linguistics: EMNLP 2020*, pages 1644–1650.
- Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213.
- David Dale, Anton Voronov, Daryna Dementieva, Varvara Logacheva, Olga Kozlova, Nikita Semenov, and Alexander Panchenko. 2021. [Text detoxification using large pre-trained neural models](#). *Preprint*, arXiv:2109.08914.
- European Commission. 2025. [The impact of the Digital Services Act on digital platforms](#). Accessed: 2025.
- European Parliament and Council of the European Union. 2022. [Regulation \(EU\) 2022/2065 of the European Parliament and of the Council of 19 October 2022 on a single market for digital services and amending Directive 2000/31/EC \(Digital Services Act\)](#). Article 17: Statement of reasons.
- Griffin Floto, Mohammad Mahdi Abdollah Pour, Parsa Farinneya, Zhenwei Tang, Ali Pesaranghader, Manasa Bharadwaj, and Scott Sanner. 2023. Diffudetox: A mixed diffusion model for text detoxification. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7566–7574.
- Jiatao Gu, Changhan Wang, and Jake Zhao. 2019. [Levenshtein transformer](#). *Preprint*, arXiv:1905.11006.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirona Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. [Deepseek-r1 incentivizes reasoning in llms through reinforcement learning](#). *Nature*, 645(8081):633–638.
- Xinlei He, Savvas Zannettou, Yun Shen, and Yang Zhang. 2023. [You only prompt once: On the capabilities of prompt learning on large language models to tackle toxic content](#). *Preprint*, arXiv:2308.05596.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. [Lora: Low-rank adaptation of large language models](#). *Preprint*, arXiv:2106.09685.
- Md Tawkat Islam Khondaker, Muhammad Abdul-Mageed, and Laks V. S. Lakshmanan. 2024. [DetoxLLM: A framework for detoxification with explanations](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 19112–19139, Miami, Florida, USA. Association for Computational Linguistics.
- Beomseok Lee, Hyunwoo Kim, Keon Kim, and Yong Suk Choi. 2024. [XDetox: Text detoxification with token-level toxicity explanations](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15215–15226, Miami, Florida, USA. Association for Computational Linguistics.
- Juncen Li, Robin Jia, He He, and Percy Liang. 2018. [Delete, retrieve, generate: A simple approach to sentiment and style transfer](#). *Preprint*, arXiv:1804.06437.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. [Understanding r1-zero-like training: A critical perspective](#). *Preprint*, arXiv:2503.20783.
- Varvara Logacheva, Daryna Dementieva, Sergey Ustyantsev, Daniil Moskovskiy, David Dale, Irina Krotova, Nikita Semenov, and Alexander Panchenko. 2022. [ParaDetox: Detoxification with parallel data](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6804–6818, Dublin, Ireland. Association for Computational Linguistics.
- Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhashkyi. 2020. [Gecor – grammatical error correction: Tag, not rewrite](#). *Preprint*, arXiv:2005.12592.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. [Training language models to follow instructions with human feedback](#). *Preprint*, arXiv:2203.02155.
- Mohammad Mahdi Abdollah Pour, Parsa Farinneya, Manasa Bharadwaj, Nikhil Verma, Ali Pesaranghader, and Scott Sanner. 2023. Count: Contrastive unlikelihood text style transfer for text detoxification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8658–8666.
- PVC (Research Infrastructure), UNSW Sydney. 2010. [Katana](#).
- Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 3982–3992.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal policy optimization algorithms](#). *Preprint*, arXiv:1707.06347.
- 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. *Preprint*, arXiv:2402.03300.

Zecheng Tang, Keyan Zhou, Juntao Li, Yuyang Ding, Pinzheng Wang, Bowen Yan, Rejie Hua, and Min Zhang. 2024. [Cmd: a framework for context-aware model self-detoxification](#). *Preprint*, arXiv:2308.08295.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. [Neural network acceptability judgments](#). *Transactions of the Association for Computational Linguistics*, 7:625–641.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. 2019. [Beyond BLEU: Training neural machine translation with semantic similarity](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4344–4355, Florence, Italy. Association for Computational Linguistics.

Jing Yu, Yibo Zhao, Jiapeng Zhu, Wenming Shao, Bo Pang, Zhao Zhang, and Xiang Li. 2025. [Text detoxification: Data efficiency, semantic preservation and model generalization](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 32172–32186, Suzhou, China. Association for Computational Linguistics.

Chiyu Zhang, Honglong Cai, Yuezhong Li, Yuexin Wu, Le Hou, and Muhammad Abdul-Mageed. 2024. [Distilling text style transfer with self-explanation from llms](#). *Preprint*, arXiv:2403.01106.

A Experimental Settings for Edit Chain Generation

In our experiments, the Editor Agent uses GPT-4o as its backbone. Within the quality gate, the semantic filter requires that the cosine similarity between source and candidate sentence embeddings from all-MiniLM-L6-v2 (Reimers and Gurevych, 2019) exceed 0.6. Of the initial 10,734 candidates, 9,095 (84.7%) pass this threshold. Zero think leaks are permitted. Maximum retry attempts: $K=3$.

B Training Hyperparameters for SFT and GRPO

In Table 5, we summarize the main training settings used in our SFT and GRPO stages for the Llama-based STEER experiments. We report both the shared optimization hyperparameters and the RL-specific settings, including group sampling, KL control, reward shaping, and granular format scoring. These values define the full training recipe underlying the main results and ablations reported in the paper.

Table 5: Training hyperparameters for SFT and GRPO stages.

Parameter	SFT	GRPO
Base model	Llama-3.1-8B-Instruct	
LoRA rank	32	32
LoRA alpha	64	64
Learning rate	1×10^{-4}	3×10^{-6}
LR schedule	Cosine	Cosine (min $0.1 \times$)
Warmup	5% of total	5% of total
Weight decay	0.01	0.01
Gradient clip	1.0	0.1
Epochs	3	3
Group size G	–	16
Temperature	–	0.70
Top- p	–	0.85
KL coefficient λ_{KL}	–	0.02
KL estimator	–	low-variance
Clip ϵ	–	0.2
Dual-clip coefficient c	–	3.0
Update epochs	–	4
<i>Reward Scorers</i>		
STA scorer	–	Cardiff offensive
SIM scorer	–	all-mpnet-base-v2
FL scorer	–	CoLA RoBERTa
<i>Veto-Aligned Reward</i>		
STA exponent α	–	0.7
SIM exponent β	–	1.3
FL exponent γ	–	0.5
FL floor τ_f	–	0.50
Baseline b	–	0.55
Scale s	–	2.5
$r_{\min}, r_{\max}$	–	–1.5, 1.0
<i>Granular Format Scoring</i>		
Base penalty r_{base}	–	–3.0
Per-element bonus δ	–	+0.3
Score ceiling r_{ceil}	–	–1.6

C In-Domain Analysis of Editing Operations and Interpretability on ParaDetox

To better understand why STEER achieves strong automatic scores, we analyze its outputs on the in-domain ParaDetox test set at the level of editing operations and reasoning traces.

As shown in Table 1, most predictions are highly localized. Among all STEER outputs, 85.8% of samples use single-step edits, with deletion (52.8%) and word substitution (34.2%) as the dominant operation types, and an average edit span of only 1.3 words. This concentration on minimal edits largely reflects the nature of ParaDetox, where toxicity is often concentrated in one short span, rather than a limitation of the framework itself.

This operation-level evidence also helps explain why STEER can surpass human references under the automatic J metric. In many cases, STEER con-

finest the change to toxic-word removal, whereas human references often make broader revisions involving punctuation, spelling, or stylistic phrasing. Such broader rewrites may be preferable from a human writing perspective, but they are not always favored by an evaluation setup that rewards strict semantic preservation and minimal intervention. We therefore interpret STEER’s advantage over the human reference in Table 1 as a consequence of benchmark preference rather than as evidence of universally better rewriting quality.

We also assess the interpretability of the generated edit chains through a human study on 200 samples, focusing on the surface consistency between the <think> reasoning and the actual edits (Table 11 in Appendix I). All dimensions score between 4.78 and 4.91 on a 1–5 scale, with moderate-to-substantial agreement (PABAK 0.46–0.77). These results suggest that STEER’s reasoning traces provide a useful post-hoc inspection record of what the model changed and why. At the same time, we emphasize that this evaluation concerns surface alignment between reasoning and edits, not causal faithfulness of the internal decision process.

D Lexicon-Based Diagnostic Slices

To test whether STEER merely performs lexical substitution, we partition the 671 ParaDetox test inputs using only source-sentence lexicon signals. C1’ contains inputs with explicit profanity or offensive lexicon hits, C3’ contains inputs with no such lexical trigger, and C4’ contains stance-bearing cases with strongly negative evaluative language but no profanity or slur-like marker. Lexicon and valence resources are used only to define diagnostic slices, not as toxicity scorers or evaluation metrics.

Table 6: Lexicon-based diagnostic slices on ParaDetox. C3’ contains no identifiable toxic lexical trigger.

Slice	N	STEER	SFT	Mask-LM	Delete
C1’ explicit	540	78.54	70.70	52.68	51.16
C3’ unmatched	94	73.77	66.48	56.62	1.06
C4’ stance	37	75.38	74.23	65.47	0.00

On C3’, where dictionary-triggered replacement has no obvious entry point, STEER still improves over Dict-Mask-LM by 17.14 J points ($p < 10^{-4}$) and over edit-chain SFT by 7.28 points ($p = 0.039$). This rules out the simplest explanation that the gains come only from lexicon-triggered deletion

or replacement, while leaving discourse-level and implicit toxicity as future work.

E Sensitivity Analysis

We first analyze the sensitivity of STEER to the SFT cold-start data proportion on ParaDetox. Table 7 shows that performance improves substantially as the SFT proportion increases from 20% to 60% and remains in a similar range thereafter. The full-data setting reaches $J = 77.7$, matching the rounded main result, while the 60% setting already reaches $J = 77.20$. These results indicate that moderate cold-start supervision provides a strong initialization for GRPO, whereas using the full filtered dataset yields the best J in this sweep. Notably, STEER with only 20% cold-start data (1,819 samples) still reaches 74.60 J , well above GEM’s 69.61 J under the same Llama backbone.

Table 7: Effect of SFT cold-start data proportion on ParaDetox. All variants undergo the same GRPO training after SFT. The full-data setting matches the rounded main result ($J = 77.7$).

Proportion	#Samples	STA	SIM	FL	J↑
20%	1,819	92.55	89.72	89.27	74.60
40%	3,638	93.59	89.52	90.16	75.72
60%	5,457	94.34	89.44	91.36	77.20
80%	7,276	94.04	89.36	91.36	77.01
100%	9,095	94.19	89.15	91.21	77.7

We then perform sensitivity analysis for the reward exponents α , β and γ used in the reward design in §3.4.1. Table 8 shows the results. As we can see that the default configuration ($\alpha=0.7$, $\beta=1.3$, $\gamma=0.5$) gives the best J . Fully linear exponents allocate less useful gradient to saturated dimensions, while overly concave settings weaken gradient signal where further improvement is possible.

Table 8: Reward exponent sensitivity on ParaDetox. “Linear” removes all concavity; “STA lin.” isolates the STA exponent; “Concave” tests aggressive concavity.

Config	α	β	γ	STA	SIM	J↑
Linear	1.0	1.0	1.0	94.0	88.5	77.1
STA lin.	1.0	1.3	0.5	95.2	89.3	76.9
Concave	0.5	1.3	0.3	93.7	89.8	76.2
STEER	0.7	1.3	0.5	<u>94.9</u>	<u>89.3</u>	77.7

F Bootstrap Significance Tests

We run 10K paired-bootstrap tests on per-example J for controlled comparisons where paired outputs

are available. As shown in Table 9, the GRPO stage significantly improves over edit-chain SFT across backbones and test sets, and the two reward-design choices also yield significant gains.

Table 9: Paired-bootstrap significance tests on per-example J . PD: ParaDetox; HF: HuggingFace; DL: DetoxLLM.

Comparison	ΔJ	95% CI	p
Llama GRPO-SFT, PD	+7.39	[+5.08,+9.72]	$< 10^{-4}$
Llama GRPO-SFT, HF	+5.31	[+2.38,+8.32]	0.0002
Llama GRPO-SFT, DL	+12.78	[+8.92,+16.69]	$< 10^{-4}$
Qwen GRPO-SFT, PD	+3.72	[+1.83,+5.62]	0.0002
Qwen GRPO-SFT, HF	+4.20	[+1.68,+6.75]	0.0006
Qwen GRPO-SFT, DL	+8.97	[+5.42,+12.67]	$< 10^{-4}$
Granular vs. flat format	+2.31	[+0.57,+4.13]	0.0098
Mult. vs. additive reward	+1.43	[+0.11,+2.73]	0.0348

G Pairwise Comparisons Derived from Likert Ratings

Table 10 presents descriptive pairwise comparisons derived from the same 200-sample Likert annotation set used in §4.5. For each sample, we first average the DET, SEM, and FLU ratings for each system and then compare systems pairwise; ties are assigned when the score difference is smaller than 0.01. These values are a transformation of the Likert ratings rather than an independently collected preference judgment.

Table 10: Pairwise comparisons on 200 ParaDetox samples derived from per-dimension Likert ratings. Win means the row-first system has a higher derived average; ties are scores within 0.01.

Comparison	Win	Tie	Lose
STEER (GRPO) vs. GEM	61.2	13.8	25.0
STEER (GRPO) vs. (SFT)	33.7	49.5	16.8
STEER (SFT) vs. GEM	58.0	13.8	28.2

H Exploratory Attempts

During development, we explored several alternatives that did not consistently improve STEER. **Token-level credit assignment.** We tried advantage reweighting, segment-level loss aggregation, compositional routing, and dual-channel reward decomposition to emphasize REWRITE tokens. This was ineffective because rewrite spans constitute only $\sim 0.5\%$ of a typical edit chain, usually contain only 1–2 tokens with a high deletion rate, and require very large amplification to affect gradient direction; such amplification destabilizes GRPO,

while post-clipping aggregation leaves high per-token variance unresolved. **Reward shaping.** Additive weighted rewards caused the policy to over-optimize style accuracy because similarity losses could be compensated by easier classifier gains. Flat penalties for malformed outputs also produced periodic zero-variance groups when many responses failed simultaneously. These observations motivated the trajectory-level multiplicative reward and granular format scoring in §3.4.1.

I Human Annotation Details for Interpretability Evaluation

Table 11 reports the second-round human annotation results for the interpretability analysis described in Appendix C. Annotators examine the generated edit chains and rate whether the `<think>` content is relevant to the subsequent edit, sufficient to justify it, and consistent with the actual rewrite. We report both mean scores and agreement statistics to quantify not only how positively the edit chains are judged, but also how consistently different annotators make these judgments.

Table 11: Interpretability annotation on 200 samples. Three annotators rate `<think>`-edit consistency on a 1–5 scale across relevance (REL), sufficiency (SUF), and consistency (CON). PABAK corrects for prevalence bias. Fleiss’ κ is near zero due to the well-known prevalence paradox: when scores concentrate at the ceiling, chance-expected agreement approaches the observed rate, deflating κ despite genuine consensus.

	REL	SUF	CON
Mean score	4.91 $\pm$.58	4.78 $\pm$.74	4.81 $\pm$.70
Raw agreement	88.3%	74.0%	72.7%
PABAK	0.766	0.480	0.455
Fleiss’ κ	−0.030	0.072	0.086