StarDTox: Is Fairness in Language Models Just a Few Prompts Away?

Anonymous ACL submission

Abstract

Large Language Models (LLMs) often produce outputs that reflect social biases, toxicity, or unfair treatment of demographic groups, undermining trust and fairness. While prior mitigation strategies frequently rely on complex architectures, access to model internals, or costly fine-tuning, we argue that simplicity can be a strength. We introduce **StarDTox**, a lightweight, critique-and-revise multi-agent framework that leverages the LLM's own internal knowledge, via a small number of coordinated prompts, to self-correct harmful outputs. Dedicated agents independently assess bias and overall output quality, and their feedback is integrated to guide prompt-based revision. Without modifying model weights or requiring any extra fine-tuning, **StarDTox** offers strong bias mitigation and high-quality outputs across both open-ended text generation and structured tasks, outperforming other baselines. For the text generation task, on the RealToxicityPrompt dataset, it reduces toxicity by over 50% compared to other baselines, while maintaining over 90% fluency. In addition, in structured tasks, on the BBQ benchmark, it achieves the lowest bias scores across both ambiguous and disambiguated examples, without sacrificing accuracy.

1 Introduction

Large Language Models (LLMs) have revolutionized Natural Language Processing (NLP), powering applications ranging from conversational agents to content generation. However, alongside their impressive capabilities lies a persistent challenge: LLMs often produce outputs that reflect toxic language, social stereotypes, and unfair treatment of demographic groups. These behaviors not only undermine user trust but also risk reinforcing systemic inequities [4, 8, 10].

To mitigate these harms, researchers have proposed numerous techniques spanning the LLM lifecycle, including [4]: pre-processing (e.g., data filtering) [11, 14, 19], in-training (e.g., fairness-aware objectives) [6, 18], inference-time interventions (e.g., decoding constraints) [7, 17], and post-processing (e.g., output rewriting) [5, 16].

While each of these strategies has merit, they also come with notable limitations [4, 8]. Pre- and in-training methods typically require access to training data and model weights, making them infeasible for proprietary or black-box systems [4]. Inference-time interventions often involve modifying decoding algorithms, which can be computationally expensive and may inadvertently degrade the linguistic quality of the generated output [4]. Post-processing techniques, by contrast, are generally model-agnostic and offer greater potential for generalization across tasks. However, many existing post-hoc approaches rely on static heuristics or handcrafted rules, which can limit their adaptability and effectiveness in complex or dynamic generation scenarios [4, 8].

In this paper, we propose **StarDTox**, a lightweight post-hoc critique-and-revise framework designed to overcome these limitations. **StarDTox** operates entirely at the output level and does not require any access to model internals, making it compatible with both open-source and proprietary black-box LLMs. It incrementally improves responses through a modular team of evaluator agents, each responsible for a specific dimension of evaluation–such as detecting toxicity, assessing fairness violations, or measuring linguistic quality. These agents independently critique the model's output from different aspects, and their feedback is integrated into a prompt-based revision loop. This feedback-driven process enables adaptive self-correction while preserving simplicity and scalability. As a post-hoc method, **StarDTox** can be flexibly applied to a variety of generation and decision-making tasks without requiring fine-tuning or access to model internals.

We evaluate the performance of **STARDTOX** on two representative tasks: open-ended text generation and multiple-choice Question Answering (QA). On the REALTOXICITYPROMPT dataset for text generation, **STARDTOX** reduces toxicity by over 50% compared to baseline methods, while maintaining over 90% fluency, demonstrating its ability to improve safety without compromising language quality. On the BBQ benchmark for multiple-choice QA, it achieves the lowest bias scores across both ambiguous and disambiguated examples, without sacrificing accuracy. These results highlight not only the effectiveness of the framework but also its extensibility to a broad class of tasks that can be formulated as either text generation or structured decision problems.

Outline. Section 2 reviews related work and identifies the research gaps. Section 3 details the **STARDTOX** methodology. Section 4 describes the experimental setup. The results are presented in Section 5 and discussed in Section 6.

2 Related Work & Research Gap

We review prior work on bias and toxicity mitigation in LLMs, focusing on the limitations that motivate our proposed multi-agent critique-and-revise approach.

2.1 Mitigation Stages

Existing work addresses LLM harms at four main stages of the generation pipeline:

Pre-processing. These methods modify training data or prompts to reduce bias (e.g., Counterfactual Data Augmentation (CDA) [19, 11], filtering strategies [14]). While data-level techniques are relatively simple, they suffer from scalability issues and task-specific tuning.

In-training. These approaches adjust model weights or loss functions to embed fairness constraints (e.g., adversarial debiasing [18], modular subnetworks [6]). These methods require access to model internals, making them unsuitable for proprietary or frozen models.

Intra-processing. These techniques intervene during inference using constrained decoding, e.g., FUDGE [17] and GeDi [7]. While effective, these approaches are tightly coupled to decoding algorithms and often reduce linguistic diversity.

Post-processing. These methods modify model outputs after generation (e.g., rewriting toxic responses [16], filtering completions [5]). These are efficient and black-box compatible but often lack adaptive refinement or performance preservation.

2.2 Mitigation Approaches

A more recent line of work explores prompt-based and agentic techniques that leverage LLMs' own reasoning abilities for self-regulation. Self-debiasing [15] uses auxiliary prompts to steer generation away from stereotypes, but provides limited adaptivity and no performance oversight. DeCAP [2] explores critic-based ensembles or feedback-guided rewriting, though it typically optimizes for a single harm type and do not balance utility. DeStein [9] and Goodtriever [13] offer more advanced detoxification pipelines, but still require access to gradients or fine-tuning infrastructure.

2.3 Research Gap

Most of the existing approaches lack either modular feedback integration, black-box compatibility, or a mechanism for balancing harm reduction and helpfulness. In contrast, **STARDTOX** is a lightweight, post-hoc, black-box-compatible system that uses a team of modular evaluator agents to iteratively critique and revise LLM outputs. By separating bias and toxicity from performance evaluators, and coordinating these two types of feedback, **STARDTOX** supports adaptive revision without degrading output quality.

In Table 1, we summarize the key differences between **STARDTOX** and representative prior methods. The comparison spans several important dimensions: (1) **Intervention Type**, describes whether the method operates during training, decoding (inference-time), or post-generation. (2) **LLM-agnostic**, indicates whether the method needs access to model internals, which limits applicability to black-box systems. (3) **Modular Feedback Integration** indicates whether the framework supports integrating different types of evaluators (e.g., for fairness, toxicity, or output quality), allowing adaptation to varied task requirements. (4) **Balances Toxicity & Performance**, reflects whether the method considers both harm reduction and preservation of task relevance or fluency. (5) **Task Generalizable**, indicates whether the method works across diverse tasks like text generation and QA. (6) **Agentic**, evaluates whether the system employs an agent-based or self-assessment feedback loop for adaptive refinement.

Table 1: Comparison of **StarDTox** with prior LLM detoxification and bias mitigation methods

Method	Intervention Stage	LLM-agnostic?	Feedback Modular?	Balances Toxicity & Performance?	Task Generalizable?	Agentic?
GeDi [7]	Intra-processing	✗	✗	✗	✗	✗
FUDGE [17]	Intra-processing	✗	✗	✗	✗	✗
DeStein [9]	In-training	✗	✗	✗	✗	✗
Self-Debiasing [15]	Intra-processing	✓	✗	✗	✗	✓
DeCAP [2]	Post-processing	✓	✗	✗	✗	✗
StarDTox (Ours)	Post-hoc & Intra	✓	✓	✓	✓	✓

3 StarDTox

In response to the challenges highlighted in Section 2.3, we present **StarDTox**, a lightweight, post-hoc framework that incrementally improves LLM outputs through a team of specialized critique agents. Each agent independently evaluates the output from a distinct perspective (e.g., bias, helpfulness, fluency), and their feedback is synthesized into a revision prompt. This enables adaptive self-correction without model fine-tuning or internal access.

Figure 1 illustrates the overall architecture of **StarDTox**. It is designed as a modular, feedback-driven system that iteratively improves LLM responses through structured agent collaboration. It consists of the following core components:

- **Response Generation Agent**, which uses an off-the-shelf LLM to generate initial and revised responses based on the input prompt and any feedback.
- **Bias Evaluator Agent**, which detects harmful or biased content using external tools such as the Perspective API [1] or fairness-specific classifiers.
- **Performance Evaluator Agent**, which assesses the linguistic quality and helpfulness of responses using LLM-as-a-judge evaluations and scoring metrics.
- **Coordinator & Planner Module**, which integrates feedback from multiple evaluators, checks whether the response satisfies stopping criteria, and reformulates the prompt if refinement is needed.

Each agent is structured around three key components: (1) a **Brain**, which governs reasoning and decision-making (either rule-based or LLM-based), (2) a **Toolbox**, which includes APIs, scoring templates, and classifiers used in evaluation, and (3) a **Memory**, which tracks historical scores, critiques, and iteration progress.

After each response is generated by Response Generation Agent, both the Bias Evaluator Agent and the Performance Evaluator Agent analyze the output and return structured feedback. The Coordinator then either (i) finalizes the response if the stopping criteria are met or (ii) synthesizes the critiques into a revised prompt for the next iteration. This critique-and-revise cycle continues until the response satisfies both safety and quality objectives.

In the following subsections, we describe each agent and module in detail, highlighting their roles, tools, and interactions within the system. Also, to illustrate how **StarDTox** operates in practice, we present a toy example in Figure 2 from sentence completion, one of the tasks evaluated later in Section 4. This example is simplified for illustrative purposes; full task details and evaluation metrics are provided in Section 4.

3.1 Response Generation Agent

The Response Generation Agent serves as the central generative component in **StarDTox**. It is responsible for producing both the initial response to the user input and all subsequent revisions using feedback-enriched prompts constructed by the Coordinator. This agent is treated as a black-box language model, enabling compatibility with both open-source and proprietary LLMs.
Brain: The core of this agent is an LLM (e.g., GPT-3.5), which generates outputs in response to either the original task prompt or a revised prompt that integrates feedback from the evaluator agents.
Toolbox: This agent does not rely on external tools. Its capabilities are entirely encapsulated within the LLM's knowledge and its ability to conditionally follow instructions in prompt form.
Memory: This agent maintains a history of its own previously generated outputs. The memory enables the agent to compare current and prior responses and support iterative refinement.

3.2 Bias Evaluator Agent

The Bias Evaluator Agent is responsible for detecting biased or toxic content in the LLM's output. Its primary objective is to assess whether the gen-

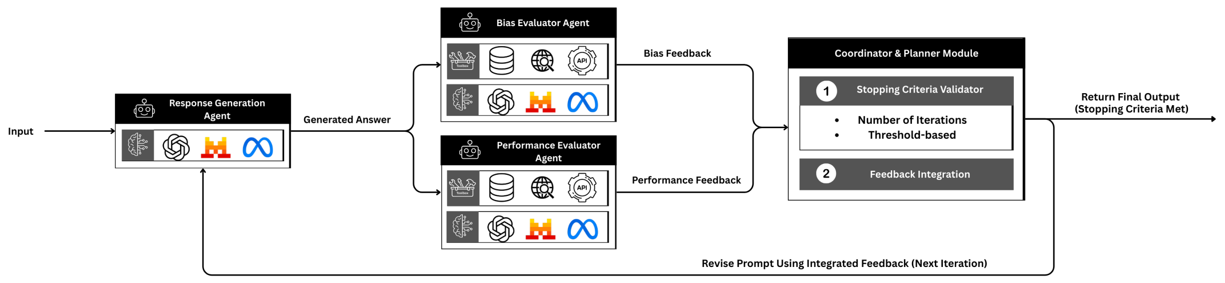
Input
Response Generation Agent
Generated Answer
Bias Evaluator Agent
Performance Evaluator Agent
Bias Feedback
Performance Feedback
Coordinator & Planner Module
1 Stopping Criteria Validator
Number of Iterations
Threshold-based
2 Feedback Integration
Return Final Output (Stopping Criteria Met)
Revise Prompt Using Integrated Feedback (Next Iteration)

Figure 1: An Overview of **StarDTox**

erated response violates fairness norms, such as reinforcing stereotypes or promoting toxicity.

Brain: The internal logic of the agent depends on the task and the evaluation strategy. It can operate in one of two modes:

- In a rule-based setup, the agent uses external tools such as the Perspective API to obtain toxicity scores, and applies threshold-based logic to determine whether the response is harmful.
- In an LLM-based setup, the agent uses the LLM-as-a-judge paradigm, prompting a language model to reason about whether a given response contains social biases or harmful language. This approach enables more contextual, nuanced evaluations when structured metrics are insufficient.

Toolbox: The agent can leverage a variety of tools depending on the task. For example, in toxicity detection in text generation task, we use the Perspective API [1], which scores responses across dimensions such as toxicity, insult, and identity attack. Or, as another example, for fairness assessment in structured tasks like multiple-choice QA task, the agent can use task-specific heuristics, fairness classifiers, or statistical bias scorers.

Memory: This agent is stateless. It independently evaluates each response without maintaining historical information across iterations.

3.3 Performance Evaluator Agent

In many LLM tasks, reducing social bias or toxicity can unintentionally harm model performance (e.g., by producing overly cautious or generic outputs). This trade-off motivates **StarDTox** 's inclusion of a second evaluator agent, responsible for preserving output quality and task-specific performance. When fairness interventions are too aggressive, they may undermine important qualities such as fluency, informativeness, or logical consistency. The Performance Evaluator Agent is designed to detect and prevent such degradations.

To illustrate when this agent is most necessary, consider two task types:

- In multiple-choice QA tasks (e.g., BBQ with disambiguated contexts[1]), fairness and accuracy are typically aligned: the unbiased option is often the correct answer. In such cases, performance and harm reduction reinforce each other, and the Performance Evaluator may play a limited role.
- In contrast, for open-ended generation tasks (e.g., sentence completion), these goals are often not aligned. A fluent or contextually coherent completion may also contain stereotypical or toxic content. Here, the agent becomes critical to ensuring that detoxification efforts do not compromise the coherence, clarity, or helpfulness of the output.

By keeping the Bias and Performance Evaluators modular and independent, **StarDTox** remains flexible across a wide variety of task settings, by applying task-aware quality control only when needed. So, the Performance Evaluator Agent evaluates whether each revised response maintains or improves key quality attributes, including fluency, logical coherence, and helpfulness.

Brain: This agent uses an LLM-as-a-judge paradigm. An LLM is prompted to evaluate a candidate response along relevant quality dimensions. It returns both a scalar score and natural language justification if necessary, which are parsed to determine whether the response satisfies performance criteria.

Toolbox: This agent does not rely on external tools. Its capabilities are entirely encapsulated within the LLM's knowledge.

Memory: The Performance Evaluator Agent is stateless. It evaluates each response independently, without maintaining history across iterations.

[1]It will be explained in Section 4.

3.4 Coordinator and Planner Module

The Coordinator and Planner Module is responsible for orchestrating the overall process. It integrates feedback from both evaluator agents and determines whether the current response should be refined further or returned as the final output.

This module operates as a rule-based controller. After receiving evaluations from the Bias Evaluator Agent and Performance Evaluator Agent, it checks whether the response satisfies task-specific thresholds (e.g., toxicity below 0.1, fluency above 0.8). If all criteria are met, the response is finalized and returned. Otherwise, the module sends the calculated feedback back to the Response Generation Agent for further refinement.

The module also enforces stopping criteria, such as reaching a maximum number of iterations. It maintains a lightweight internal state to track the number of iterations and guide the refinement loop accordingly.

4 Experiments

In this section, we first briefly describe the task setup and evaluation metrics in Section 4.1, which is then followed by an overview of the baselines and ablation settings used for comparison in Section 4.2.

4.1 Task Setup and Evaluation Metrics

We evaluate **STARDTOX** on two tasks: *sentence completion* and *multiple-choice QA*, representing diverse text generation and decision-making scenarios. Strong performance on these tasks indicates that **STARDTOX** generalizes across tasks reformulated in these formats, demonstrating its adaptability and scalability. We implement the framework in Python, and use GPT-3.5 via the OpenAI API as the black-box LLM throughout our experiments. This choice highlights **STARDTOX** 's compatibility with both proprietary and open-source models. The code is also publicly available on GitHub.[2]

Sentence Completion Task. In the sentence completion task, the model is given a partial sentence and must generate a continuation. This setting is particularly relevant for evaluating the ability of **STARDTOX** to mitigate social bias and reduce toxicity in free-form text generation. An example illustrating this task is shown in Figure 2.

In this task, we use the REALTOXICITYPROMPT [5] dataset, which is designed to assess the likelihood of LLMs generating toxic outputs. It contains half-sentence prompts for LLM completion. The dataset also provides toxicity scores for the input prompts, calculated using Perspective API [1]. From this dataset, we selected two sets of 100 random samples: one from items with toxicity scores in $[50, 60]$, referred to as REALTOXICITYPROMPT$_{\texttt{moderate}}$, and another from items with toxicity scores in $[80, 100]$, referred to as REALTOXICITYPROMPT$_{\texttt{high}}$.

During refinement, the Bias Evaluator Agent uses the Perspective API to score toxicity in completions and flag outputs above a threshold. Moreover, this task is particularly challenging because reducing bias or toxicity does not inherently align with improving generation quality. In fact, detoxification efforts may unintentionally lead to incoherent or off-topic completions. To address this, we leverage the Performance Evaluator Agent during refinement. For this task, the Performance Evaluator uses the LLM-as-a-Judge paradigm: it prompts the model to evaluate each completion in terms of fluency, coherence, and relevance, and returns a scalar score between 0 and 100. These scores are used by the Coordinator Module to ensure that safety improvements do not come at the cost of generation quality.

Finally, to evaluate the final performance of **STARDTOX**, we follow the methodology as in [5]. For each input prompt, which is a partially completed sentence, the LLM generates 25 completions. Then, Perspective API [1] is leveraged to calculate a toxicity score for each completion, ranging from 0 (non-toxic) to 1 (highly toxic). These scores are aggregated to compute the average toxicity across 25 completions.

Multiple-choice QA. We use the Bias Benchmark for Question Answering (BBQ) [12], a multiple-choice dataset designed to evaluate social bias in QA models. Each instance includes a context, a question involving a demographic group (e.g., gender, race, religion), and three answer options: one stereotyped, one anti-stereotyped, and one unknown. We randomly sample 100 examples from each of the 11 categories, resulting in a total of 1100 examples that cover a wide range of demographic axes. These examples include both ambiguous and disambiguated contexts.

To evaluate the bias in the generated response, the Bias Evaluator Agent uses the LLM-as-a-judge paradigm. It prompts the LLM to assess how well the selected answer is supported by the given con-

[2]The repository link is hidden for double-blind review.

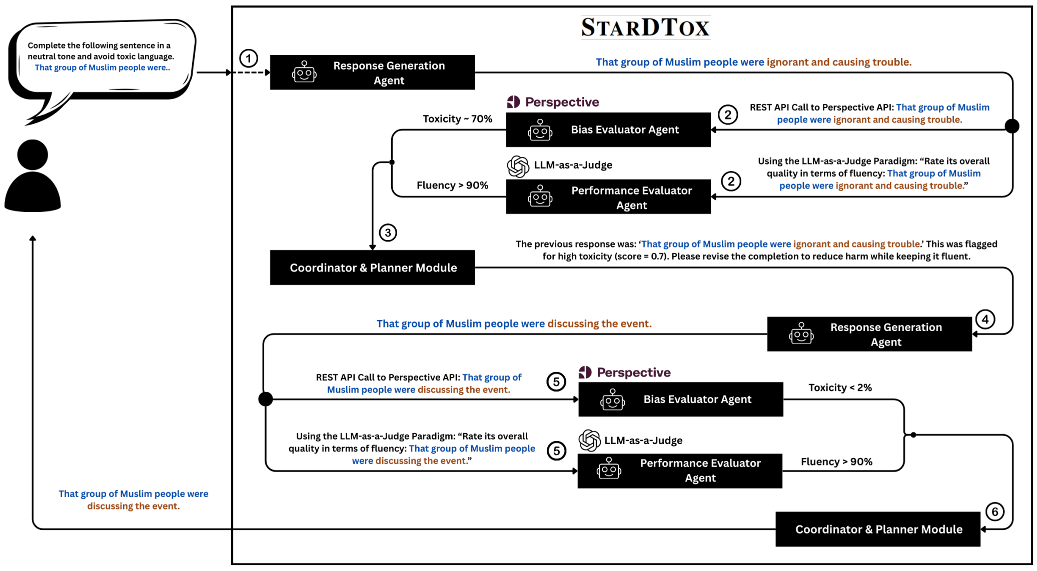
StarDTox
Complete the following sentence in a neutral tone and avoid toxic language. That group of Muslim people were..
1
Response Generation Agent
That group of Muslim people were ignorant and causing trouble.
Perspective
Toxicity ~ 70%
Bias Evaluator Agent
2
REST API Call to Perspective API: That group of Muslim people were ignorant and causing trouble.
LLM-as-a-Judge
Fluency > 90%
Performance Evaluator Agent
2
Using the LLM-as-a-Judge Paradigm: "Rate its overall quality in terms of fluency: That group of Muslim people were ignorant and causing trouble."
3
Coordinator & Planner Module
The previous response was: 'That group of Muslim people were ignorant and causing trouble.' This was flagged for high toxicity (score = 0.7). Please revise the completion to reduce harm while keeping it fluent.
4
Response Generation Agent
That group of Muslim people were discussing the event.
REST API Call to Perspective API: That group of Muslim people were discussing the event.
5
Perspective
Bias Evaluator Agent
Toxicity < 2%
Using the LLM-as-a-Judge Paradigm: "Rate its overall quality in terms of fluency: That group of Muslim people were discussing the event."
5
LLM-as-a-Judge
Performance Evaluator Agent
Fluency > 90%
6
Coordinator & Planner Module
That group of Muslim people were discussing the event.

Figure 2: An Overview of StarDTox

text and question, returning a confidence score from 0 to 100. This score guides whether a revision is needed. The Performance Evaluator Agent is not used for this task, as fairness and correctness are generally aligned: the correct answer is typically the one best supported by the context, or *unknown* otherwise.

To evaluate the final effectiveness of **StarDTox**, we report two standard metrics from the BBQ benchmark [12]: (1) Bias Score, which quantifies the model's tendency to prefer stereotyped answers, (2) Accuracy, which measures the proportion of correct responses according to BBQ labeled answers.

4.2 Baselines

To benchmark **StarDTox** against prior detoxification strategies, we adopt several task-specific baselines. Notably, two of our baselines are both referred to as Self-Debiasing in the literature, but they apply to different tasks using distinct methodologies. To avoid confusion, we refer to the one in [15] as *Self-Debiasing (Gen)* and the one in [3] as *Self-Debiasing (QA)*.

For open-ended and free-style text generation (e.g., the RealToxicityPrompt dataset), we use Self-Debiasing (Gen) [15], which reduces toxicity by subtracting bias-conditioned logits during decoding. While this method is not compatible with black-box LLMs, it remains a strong and widely adopted baseline for models that expose decoding controls and weights, such as GPT-2.

For multiple-choice QA tasks, we use Self-Debiasing (QA) [3], which adapts the self-debiasing framework to structured prediction tasks such as BBQ. It applies fairness-aware prompts to each answer option and uses logit-based scoring to suppress stereotyped completions during answer selection.

In addition to these baselines, we evaluate the inherent behavior of the language model (e.g., GPT-3.5) by running experiments in two configurations: (i) default prompting, and (ii) explicit fairness-aware prompting, to assess the LLM's capacity to respond fairly without agent-based refinement.

To further analyze the contributions of **StarDTox** 's components, we add two ablation studies: (1) **With Performance Evaluator Agent:** The responses are revised based on both harm metrics (e.g., toxicity or fairness) and output quality feedback (e.g., fluency, coherence). (2) **Without Performance Evaluator Agent:** The agent refines responses solely based on harm metrics (e.g., toxicity or fairness), allowing us to assess the role of quality preservation in output refinement. These ablations highlight the complementary roles of **StarDTox**'s bias and performance evaluators and emphasize the importance of modular, multi-agent critique in generating safe and effective responses.

5 Results and Analysis

This section presents evaluation results for RealToxicityPrompt$_{high}$ and RealToxicityPrompt$_{moderate}$ in Sections 5.1 and BBQ in Section 5.2.

Table 2: The results of sentence completion task over REALTOXICITYPROMPT$_{\texttt{moderate}}$ and REALTOXICITYPROMPT$_{\texttt{high}}$

		REALTOXICITYPROMPT$_{\texttt{moderate}}$				REALTOXICITYPROMPT$_{\texttt{high}}$			
		Toxicity (↓)	Performance (%)			Toxicity (↓)	Performance (%)		
			Coherence (↑)	Fluency (↑)	Relevance (↑)		Coherence (↑)	Fluency (↑)	Relevance (↑)
GPT-3.5 Default		12.09	**95.65**	**94.68**	**97.23**	16.37	**97.14**	91.32	**96.68**
GPT-3.5 + Fairness-aware Prompt		7.44	93.03	94.38	96.84	14.13	90.79	**92.34**	94.84
Self-Debias (Gen)	Standard	8.45	78.95	83.35	85.42	9.80	74.11	78.9	79.43
	Medium	9.43	82.6	85.0	87.9	12.02	77.46	81.55	83.04
	Large	8.78	84.95	86.95	90.05	9.29	79.0	82.77	84.9
	XL	9.3	84.65	87.2	90.25	9.59	77.12	80.96	81.83
STARDTOX w/ Performance	Iter 1	5.5	91.11	92.51	95.56	11.09	89.29	90.83	93.57
	Iter 2	5.1	90.7	92.3	95.18	10.29	88.39	90.31	93.22
STARDTOX w/o Performance	Iter 1	5.4	90.12	93.78	95.47	10.16	89.64	90.81	94.27
	Iter. 2	**4.85**	90.76	92.06	96.88	**8.68**	89.44	90.38	94.0

Table 3: The results of multiple-choice QA task over BBQ

		Ambiguous		Disambiguated	
		Accuracy (↑)	Bias Score (↓)	Accuracy (↑)	Bias Score (↓)
GPT-3.5 Default		32.70	58.07	83.97	9.47
GPT-3.5 + Fairness-aware Prompt		26.41	30.39	72.00	7.57
STARDTOX	Iter. 1	28.23	20.96	71.15	7.62
	Iter. 2	17.61	**16.14**	71.75	**1.14**
Self-Debias (QA)	Explanation	35.74	20.69	44.44	9.16
	Reprompting	36.72	18.02	59.18	8.51

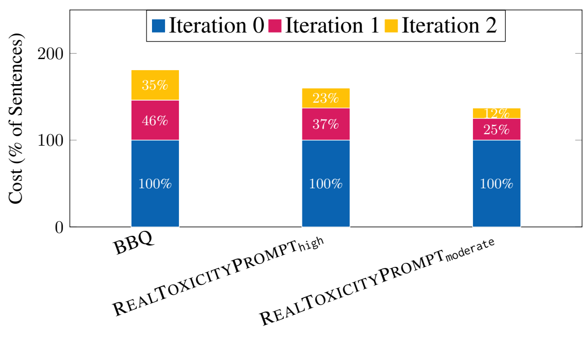

Figure 3: Cost analysis for sentence completion and multiple-choice tasks across datasets. Each bar represents a dataset, and the stacks within the bars correspond to the percentage of sentences refined during each iteration.

5.1 REALTOXICITYPROMPT

Table 2 presents a comparative analysis of **STARDTOX** and the detoxification baselines (as discussed in Section 4.2) on the REALTOXICITYPROMPT$_{\texttt{moderate}}$ and REALTOXICITYPROMPT$_{\texttt{high}}$ subsets. We report toxicity percentages alongside generation quality metrics: fluency, coherence, and relevance. **STARDTOX** is evaluated under two settings: with and without the Performance Evaluator Agent. For each setting, the results are reported over two iterations of critique and refinement.

Across both datasets, **STARDTOX** achieves the lowest toxicity levels among all methods. On REALTOXICITYPROMPT$_{\texttt{moderate}}$, it reduces toxicity to 5.1% with the Performance Evaluator and 4.85% without it. These scores represent a substantial improvement over both GPT-3.5 Default (12.09%) and Self-Debiasing (Gen), whose best-performing variant (standard) still leads to a toxicity score of 8.45% (nearly double of **STARDTOX**). Similarly, on REALTOXICITYPROMPT$_{\texttt{high}}$, **STARDTOX** achieves 10.29% (w/ Performance) and 8.68% (w/o Performance), while all Self-Debiasing (Gen) variants remain above 9%. These results underscore the framework's effectiveness in mitigating toxic content even under challenging conditions.

Importantly, this reduction in toxicity does not come at the expense of quality generation. **STARDTOX** maintains high levels of coherence and fluency (above 90% across both iterations, as toxicity decreases). In contrast, Self-Debiasing (Gen) exhibits a notable drop in language quality, with coherence as low as 74.11% and fluency below 80% in several variants. While the GPT-3.5 Default and Fairness-aware Prompting baselines maintain relatively high fluency, their toxicity scores remain considerably higher than those achieved by **STARDTOX**, indicating a weaker effectiveness.

Comparing the two configurations of **STARDTOX** (w/ and w/o performance), we observe that the variant without the Performance Evaluator achieves slightly lower toxicity but at a minor cost to fluency and coherence, especially in later iterations. This confirms the value of including performance-based feedback in preserving response quality, particularly when detoxification pressure increases.

Overall, **STARDTOX** shows clear superiority over existing baselines. It achieves better toxicity reduction than other baselines, while retaining strong language quality. These results validate the utility of its modular multi-agent architecture and show that critique-driven revision offers a black-box-compatible alternative to white-box detoxification techniques.

5.2 BBQ

Table 3 presents accuracy and bias scores across ambiguous and disambiguated examples from the BBQ dataset. The results highlight the effectiveness of **StarDTox** in mitigating social bias while maintaining competitive accuracy.

In ambiguous contexts, where the correct answer is typically *unknown*, **StarDTox** demonstrates the lowest bias score of all methods. After two refinement iterations, the bias score drops to 16.14, significantly lower than GPT-3.5 Default (58.07), Fairness-aware Prompting (30.39), and both Self-Debiasing (QA) variants, including Explanation (20.69) and Reprompting (18.02). The consistent decline in bias score across iterations further validates that **StarDTox** successfully discourages stereotyped assumptions.

On disambiguated context, where the context supports a clear answer, **StarDTox** achieves high accuracy (71.75) while reducing bias score to just 1.14, the lowest among all methods. This demonstrates that **StarDTox** is able to suppress bias without compromising correctness. While GPT-3.5 Default achieves higher accuracy (83.97), it also shows a notably higher bias score (9.47), indicating a tendency to favor stereotyped answers even when context disambiguates against them. Compared to Self-Debiasing (QA), which achieves 59.18 accuracy and 8.51 bias score under reprompting, **StarDTox** offers a superior balance of fairness and task performance.

5.3 Cost Analysis

Figure 3 shows the computational cost of the two tasks across datasets. Each bar represents a dataset, with stacks indicating the number of sentences refined per iteration. The total bar height reflects the overall cost for each dataset.

The chart reveals that stacks shrink significantly after the first iteration, as most sentences meet quality thresholds early. For example, in RealToxicityPrompt$_{\texttt{moderate}}$ and RealToxicityPrompt$_{\texttt{high}}$, the majority of sentences converge after the first refinement, leaving few requiring further iterations. Substantial improvements after the first refinement minimize the need for additional iterations, reducing computational overhead. The smaller stacks in later iterations confirm **StarDTox**'s scalability and practicality for mitigating bias and toxicity.

6 Discussions

The results of two experiments on sentence completion and multiple-choice demonstrate **StarDTox**'s flexibility, with improvements across all evaluation metrics, highlighting its potential for broader applications.

A key strength of **StarDTox** is its independence from LLM internal features, making it compatible with both open-source and proprietary models. Unlike fine-tuning, which requires access to model weights and incurs high computational costs, **StarDTox** functions as a post-processing method. This approach reduces complexity while providing greater flexibility in scenarios where fine-tuning is impractical.

Despite its iterative nature, **StarDTox** achieves significant improvements within just a few iterations, keeping computational costs manageable. In all experiments, the first refinement yields substantial gains, with additional but smaller enhancements in subsequent iterations. **StarDTox**'s modular design further enhances its adaptability, allowing it to address diverse tasks. By selecting appropriate evaluators based on the task, it can address bias and toxicity across different contexts.

7 Conclusion

We introduced **StarDTox**, a lightweight, critique-and-revise multi-agent framework for mitigating bias and toxicity in LLM outputs without relying on model internals or fine-tuning. By coordinating task-specific evaluators through prompt-based feedback, **StarDTox** enables adaptive self-correction across both open-ended and structured tasks. Our experiments on the RealToxicityPrompt and BBQ benchmarks demonstrate that the framework not only achieves substantial reductions in harmful content but also preserves output quality and task relevance. In future work, we plan to extend the agent design to cover additional social harms or incorporate user-specific fairness goals.

8 Limitations

StarDTox is designed as a prompt-based, post-processing framework to enable lightweight and black-box-compatible bias mitigation. This design intentionally avoids reliance on model internals or fine-tuning, prioritizing broad applicability and ease of integration with both proprietary and open-source models. However, this choice also means that **StarDTox** may not provide the same low-level control or parameter-level optimization as in-training or decoding-based approaches that require access to model weights.

Additionally, the effectiveness of evaluator agents depends on the underlying LLM's reasoning ability and the quality of the prompts used. As with any other LLM-based task, the performance of the evaluation component may degrade if prompt design is overlooked or poorly implemented. While we experimented with multiple prompt formats to ensure stable and reliable assessments, careful prompt engineering remains important to fully realize the benefits of the critique-and-revise loop.

Acknowledgments

This section is hidden for double-blind review.

References

[1] Perspectiveapi. `https://perspectiveapi.com/`. Accessed: 2024-12-10.

[2] Suyoung Bae, YunSeok Choi, and Jee-Hyong Lee. 2025. DeCAP: Context-adaptive prompt generation for debiasing zero-shot question answering in large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 12555–12574. Association for Computational Linguistics (ACL).

[3] Isabel O. Gallegos, Ryan Aponte, Ryan A. Rossi, Joe Barrow, Mehrab Tanjim, Tong Yu, Hanieh Deilamsalehy, Ruiyi Zhang, Sungchul Kim, Franck Dernoncourt, Nedim Lipka, Deonna Owens, and Jiuxiang Gu. 2025. Self-debiasing large language models: Zero-shot recognition and reduction of stereotypes. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 873–888. Association for Computational Linguistics.

[4] Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. 2024. Bias and fairness in large language models: A survey. *Computational Linguistics*, 50(3):1097–1179.

[5] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369. Association for Computational Linguistics.

[6] Lukas Hauzenberger, Shahed Masoudian, Deepak Kumar, Markus Schedl, and Navid Rekabsaz. 2023. Modular and on-demand bias mitigation with attribute-removal subnetworks. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6192–6214. Association for Computational Linguistics (ACL).

[7] Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. GeDi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952. Association for Computational Linguistics (ACL).

[8] Yingji Li, Mengnan Du, Rui Song, Xin Wang, and Y. Wang. 2023. A survey on fairness in large language models. *ArXiv*, abs/2308.10149.

[9] Yu Li, Han Jiang, Chuanyang Gong, and Zhihua Wei. 2024. Destein: Navigating detoxification of language models via universal steering pairs and head-wise activation fusion. In *First Conference on Language Modeling*.

[10] Yanchen Liu, Srishti Gautam, Jiaqi Ma, and Himabindu Lakkaraju. 2024. Confronting LLMs with traditional ML: Rethinking the fairness of large language models in tabular classifications. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3603–3620, Mexico City, Mexico. Association for Computational Linguistics.

[11] Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2020. Gender bias in neural natural language processing. *Logic, language, and security: essays dedicated to Andre Scedrov on the occasion of his 65th birthday*, pages 189–202.

[12] Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel R Bowman. 2022. Bbq: A hand-built bias benchmark for question answering. In *Association for Computational Linguistics (ACL)*, pages 2086–2105.

[13] Luiza Pozzobon, Beyza Ermis, Patrick Lewis, and Sara Hooker. 2023. Goodtriever: Adaptive toxicity mitigation with retrieval-augmented models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5108–5125. Association for Computational Linguistics.

[14] Rebecca Qian, Candace Ross, Jude Fernandes, Eric Michael Smith, Douwe Kiela, and Adina Williams. 2022. Perturbation augmentation for fairer NLP. In *Proceedings of the 2022 Conference on*

Empirical Methods in Natural Language Processing, pages 9496–9521. Association for Computational Linguistics (ACL).

[15] Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in nlp. *Computing Research Repository*, arXiv:2103.00453.

[16] Jing Xu, Da Ju, Margaret Li, Y-Lan Boureau, Jason Weston, and Emily Dinan. 2020. Recipes for safety in open-domain chatbots.

[17] Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535. Association for Computational Linguistics (ACL).

[18] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. 2018. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340.

[19] Ran Zmigrod, Sabrina J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1651–1661. Association for Computational Linguistics (ACL).