Reproducibility study of FACTER: Fairness-Aware Conformal Thresholding and Prompt Engineering

Anonymous authors
Paper under double-blind review

Abstract

We present a reproducibility study of FACTER, a post-hoc framework that combines conformal thresholding with iterative prompt engineering to mitigate demographic bias in black-box LLM-based recommender systems. Using the released codebase and experimental setting from the original paper, we evaluate FACTER on MovieLens-1M and Amazon Movies & TV with various LLM backbones. We assess fairness using the reported violation-based criterion group and counterfactual metrics (SNSR, CFR), and measure recommendation quality via catalog-mapped ranking metrics (NDCG@10, Recall@10) alongside the validity rate of generated items (Valid@10). Across datasets and supported backbones, we reproduce FACTER's key qualitative behavior, with fairness violations decreasing sharply and converging within a small number of calibration rounds. However, unlike the original study, we observe a collapse in recommendation quality, largely driven by low validity of generated movie titles under open-vocabulary generation, inaccurate item-mapping and evaluation assumptions. We further identify and resolve multiple implementation and reproducibility issues in the released code, providing a cleaner and easier-to-run codebase to support future replication. Overall, our findings support FACTER's effectiveness in reducing measured fairness violations, but with a higher loss in utility.

1 Introduction

Large Language Models (LLMs) have rapidly become a central component of modern decision-support systems, extending beyond classical natural language processing tasks. Due to their ability to function as zero-shot or prompt-based recommender systems, they can be used for a wide range of applications without task-specific fine-tuning (Zhang et al., 2023; Hou et al., 2024), including recommendation (Zhang et al., 2023; Hou et al., 2024) and candidate screening or hiring-related decision support (Drushchak & Romanyshyn, 2024). This flexibility enables rapid deployment across a wide range of settings, positioning LLMs as attractive alternatives to traditional recommendation pipelines.

At the same time, LLM-based recommendation raises significant fairness concerns (Zhang et al., 2023; Sharma et al., 2023). LLMs can encode and amplify societal biases related to gender, age, or race even when such attributes are only minimally varied in the input (Sheng et al., 2019; Blodgett et al., 2020), producing systematic differences in content, tone, or exposure rather than explicit discriminatory language (Sheng et al., 2019; Lucy & Bamman, 2021; Zhao et al., 2018). These subtle biases are particularly difficult to detect, yet can carry meaningful downstream consequences, especially in high-stakes domains like hiring (Dwork et al., 2012; Bender et al., 2021).

However, improving fairness is not only a matter of reducing demographic disparities. In recommender systems, fairness interventions must also preserve sufficient recommendation quality, since users may abandon a system if its outputs no longer reflect their preferences or provide useful recommendations. This creates a central fairness–utility trade-off: stronger fairness constraints can reduce biased behavior, but may also lower predictive performance or recommendation relevance (Dwork et al., 2012; Madras et al., 2018). Conversely, optimizing only for utility may preserve user satisfaction in the short term while allowing systematic demographic disparities to persist. The practical value of a fairness-aware recommender depends not only on whether it reduces unfair outcomes, but also on an acceptable cost to utility.

Unlike traditional machine learning models, many state-of-the-art LLMs are deployed as black-box systems through commercial APIs, offering no access to training data, intermediate representations, or model parameters (Touvron et al., 2023; Grattafiori et al., 2024). As a result, many established fairness interventions that rely on retraining, architectural modification, or direct control over the learning process are infeasible in practice (Madras et al., 2018; Sun et al., 2019). This limitation motivates the need for fairness mechanisms that can operate post-hoc and without access to model internals, particularly in black-box LLM deployments where intervention must happen entirely at inference time (Fayyazi et al., 2025).

In response to these constraints, Fayyazi et al. (2025) propose FACTER, a fairness-aware framework for mitigating demographic bias in LLM-based recommender systems. While prior work has explored fairness interventions through model retraining or representation learning (Madras et al., 2018; Sun et al., 2019), and prompt-based steering more broadly has been studied as a practical way to control black-box LLM behavior (Reynolds & McDonell, 2021), FACTER focuses specifically on a fully post-hoc fairness setting where the underlying model cannot be modified. In contrast to approaches that depend on changing model parameters or fine-tuning with additional fairness objectives, FACTER combines conformal prediction with iterative prompt engineering to detect fairness violations and adapt the prompt accordingly. Conformal prediction Shafer & Vovk (2008) is a distribution-free statistical framework that uses a held-out calibration set to set a threshold below which a specified fraction of normal outputs fall, ensuring that the false-positive rate of flagging new observations as anomalous is bounded by a user-specified level α, under exchangeability assumptions. FACTER uses this to address a fundamental challenge in fairness auditing: determining when a semantic difference between recommendations for different demographic groups is large enough to constitute a violation. By fitting the threshold to the empirical distribution of fairness scores on calibration data, rather than setting it arbitrarily, FACTER ensures that the probability of falsely flagging a fair recommendation as biased remains statistically controlled.

This approach makes the framework particularly relevant for deployed LLM recommenders accessed only through inference-time interaction. The original paper reports that, across MovieLens and Amazon datasets, FACTER substantially reduces fairness violations while largely preserving recommendation accuracy, suggesting a favorable balance between fairness and utility in both dense and sparse data regimes (Fayyazi et al., 2025). This claim is central to the contribution of the original paper: if fairness can be improved without substantially lowering utility, then FACTER would offer a practical mechanism for mitigating demographic bias in deployed black-box recommendation systems, as it is a post-hoc model-agnostic approach.

This paper presents a reproducibility study of FACTER. Our motivation stems from the fact that FACTER targets black-box LLM deployments, where model internals and training data are inaccessible and mitigation must rely entirely on external auditing and prompt-based interventions. In such settings, the validity of fairness guarantees depends not only on the high-level method, but also on the robustness of the surrounding evaluation pipeline, including data preprocessing, output parsing, item mapping, and metric computation. Reproducibility is therefore particularly important: if these components are underspecified, the reported fairness–utility trade-offs may be difficult to verify in practice.

Therefore, in this work we reproduce FACTER and evaluate whether its main empirical claims hold under the released code and experimental setup. Our contributions are fourfold. First, we identify and solve implementation gaps, missing details, and paper–code mismatches that materially affect the reproducibility of the framework. Second, we show that the main qualitative fairness result is reproducible, as FACTER consistently reduces measured fairness violations. However, we find that this reduction is accompanied by a much larger drop in recommendation utility than reported in the original study. Third, we provide additional analysis of the evaluation pipeline, showing that low recommendation validity and fragile title-to-catalog mapping play a central role in the observed discrepancy. Finally, we release a cleaner and easier-to-run version of the codebase to support future work on post-hoc fairness in LLM-based recommendation.

2 Scope of reproducibility

Using the same experimental setting as the original paper (datasets, LLM backbones, baselines, and evaluation metrics), we assess whether the main empirical claims of Fayyazi et al. (2025) can be reproduced using the released code and setup. Based on the original paper, we identify the following testable claims:

- **Claim 1:** *FACTER substantially reduces fairness violations relative to the initial/Zero-Shot setting across different LLM backbones, while maintaining comparable recommendation accuracy.* The authors report that "*Empirical results on MovieLens and Amazon show that FACTER substantially reduces fairness violations (up to 95.5%) while maintaining strong recommendation accuracy.*"
- **Claim 2:** *FACTER is model-agnostic, consistently reducing fairness violations across different LLM backbones.* The paper reports that all models show monotonic improvement, with fairness violations reduced by at least *90%* across LLaMA-3-8B, LLaMA-2-7B, and Mistral-7B.
- **Claim 3:** *FACTER exhibits monotonic convergence behavior across calibration iterations.* As stated by the authors, "*all LLMs exhibit a monotonic improvement, with LLaMA3-8B converging near zero violations by Iteration 3.*"
- **Claim 4:** *FACTER remains effective in sparse data regimes by reducing fairness violations while preserving recommendation performance relative to baselines.* When evaluated on the Amazon Movies & TV dataset, the authors state that "*despite the greater sparsity, our approach still reduces violations substantially (a 90.9% drop),*" suggesting that both fairness improvements and utility are maintained even under sparse interaction settings.

A point of ambiguity in the original paper concerns the iteration numbering used to report violation reduction. Table 1 of the original study compares FACTER directly against the Zero-Shot baseline, while their Figure 3 shows FACTER trajectory from Iteration 0 to Iteration 3 and their Figure 4 shows model-wise trajectories from Iteration 1 to Iteration 3. Since both figures describe the same type of iterative violation reduction, Figure 4 appears to be offset by one iteration relative to Figure 3. We therefore interpret the initial FACTER state as the unmitigated starting point, corresponding most closely to the Zero-Shot setting.

3 Methodology

3.1 FACTER

Core intuition. FACTER is based on the idea that unfairness in LLM-based recommendation often manifests as semantic instability: when two users with identical interaction histories but different protected attributes receive substantially different recommendations. If such differences exceed what is considered normal variation, the recommendation is flagged as unfair. FACTER formalizes this notion using embedding-based distance measures and controls the rate of detected violations through conformal prediction.

Operational loop. At a high level, FACTER runs as a closed-loop system with four recurring steps: (i) the LLM generates a ranked recommendation list for a given user context; (ii) the output is audited for fairness by comparing it to counterfactual outputs with flipped protected attributes; (iii) if a fairness violation is detected, the system records the violation and updates its fairness constraints; and (iv) these constraints are injected into subsequent prompts to discourage repeated demographic biases.

Recommendation generation and counterfactual auditing. Formally, given a user context x (e.g., an interaction history) and a protected-attribute vector a (e.g., gender, age, occupation), the LLM produces a ranked recommendation list $\hat{y} = \hat{Y}(x, a)$. To audit fairness, FACTER generates counterfactual outputs by varying only the protected attributes while keeping x fixed, and evaluates how much the resulting recommendations diverge in semantic space.

Fairness-aware nonconformity score. Each recommendation is assigned a nonconformity score:

$$S_i = d_i + \lambda \Delta_i, \tag{1}$$

which combines predictive accuracy and fairness. The term d_i measures how far the generated recommendation deviates from a reference item in embedding space:

$$d_i = 1 - \cos\big(\mathrm{Emb}(\hat{y}_i), \mathrm{Emb}(y_i)\big), \tag{2}$$

where Emb(·) denotes a sentence embedding model and y_i is the reference (ground-truth) item.

The fairness penalty Δ_i captures cross-group semantic disparity:

$$\Delta_i = \max_{j:W_{ij}>\tau_\rho} \|\text{Emb}(\hat{y}_i) - \text{Emb}(\hat{y}_j)\|_2 \,, \tag{3}$$

where W_{ij} measures similarity between user contexts, τ_ρ restricts comparisons to similar contexts, and only pairs with differing protected attributes are considered. The hyperparameter $\lambda > 0$ controls the trade-off between fairness and recommendation accuracy.

Offline calibration. To decide when a semantic deviation constitutes a fairness violation, FACTER relies on conformal prediction. A threshold $Q_\alpha^{(t)}$ is calibrated on a held-out dataset such that, under exchangeability assumptions, the probability of falsely flagging a recommendation remains bounded by α. If a generated recommendation yields $S_i > Q_\alpha^{(t)}$, it is labeled as a fairness violation.

Online prediction. When violations occur, FACTER adapts its fairness constraints in two ways. First, the conformal threshold is updated using the rule implemented in the released code: We follow the exact formulation provided in the original implementation, where the threshold is updated as

$$Q_\alpha^{(t+1)} = \gamma Q_\alpha^{(t)} + (1-\gamma)\min\left(Q_\alpha^{(t)}, S_i\right), \tag{4}$$

where $\gamma \in (0,1)$ controls how quickly the system reacts to new observations. Notably, under this formulation, if a violation occurs (i.e., $S_i > Q_\alpha^{(t)}$), the update simplifies to $Q_\alpha^{(t+1)} = Q_\alpha^{(t)}$, meaning that the threshold is not tightened in response to individual violations. This behavior differs from the intended interpretation described in the original paper, where the threshold is expected to adapt when violations are detected.

The rationale for this adaptive update is that a static threshold, although calibrated offline, may become insufficiently strict as new violation patterns emerge during deployment. However, in practice, threshold adaptation is primarily driven by the accumulation of violations rather than single events.

Second, recurring violation patterns are distilled into group-specific prompt instructions, which are appended to the system prompt. Formally, let V denote a buffer of past violations. For a given protected attribute value a, the system filters all violations in V with the same attribute and extracts salient features (e.g., genre or content patterns) from the corresponding outputs. If a feature f appears at least k times (with $k \geq 3$ in the implementation), the system generates an instruction of the form

$$I^{(t+1)} = I^{(t)} \cup \{\text{“Avoid: } (a) \rightarrow f\text{-only”}\}, \tag{5}$$

which discourages the model from producing recommendations dominated by that feature for the given group. This mechanism ensures that prompt updates are triggered only after repeated violations, reducing sensitivity to noise. A concrete example of such prompt updates is provided in Appendix D.

Overall, FACTER provides a lightweight, fully post-hoc mechanism for mitigating demographic biases in LLM-based recommender systems by combining conformal thresholding with adaptive prompt engineering, while preserving the underlying model and its deployment constraints.

3.2 Datasets

We evaluate FACTER on the original paper's datasets across dense and sparse recommendation settings.

MovieLens-1M [1] dataset (Harper & Konstan, 2015) consists of approximately one million user–item interactions collected from the MovieLens online recommender system over several years. Each interaction corresponds to a user rating a movie, accompanied by user metadata such as gender, age group, and occupation. Following the original paper, MovieLens-1M serves as the primary benchmark for evaluating fairness behavior in relatively dense recommendation settings. To further characterize the demographic structure of

[1] MovieLens-1M: `https://files.grouplens.org/datasets/movielens/ml-1m.zip`

the dataset used in our experiments, we compute descriptive statistics over protected attributes, including occupation distributions, joint gender-age-occupation group counts, and the preservation of group proportions across calibration and test splits. These statistics are reported as figures in Appendix B.

Amazon Movies & TV [2] dataset is derived from the larger Amazon product interaction corpus introduced by McAuley et al. (2015). It contains user-item interactions with randomly assigned sensitive attributes, in a substantially sparser setting than MovieLens, with a long-tailed item distribution and fewer interactions per user. This dataset is used to assess the robustness of FACTER under sparse data conditions, where recommendation accuracy is more challenging and fairness violations may be harder to detect. Descriptive statistics can be found in Appendix B.

Data sampling follows the calibration-test split of 70:30 of the original set-up. However, the original paper does not provide details on how samples are drawn from the datasets or whether stratification is applied. To resolve this ambiguity, we adopted random sampling with stratification over protected attributes to preserve group proportions across splits. This choice is consistent with the fairness evaluation setting and helps prevent unintended distributional shifts between calibration and test data. While alternative sampling strategies are possible, we found that unstratified sampling could lead to unstable fairness metrics for underrepresented groups, particularly in the sparse Amazon dataset. This stratified sampling procedure was applied consistently to both datasets.

3.3 Hyperparameters

We set hyperparameters based on the original paper and the the authors' final public code repository [3]. Table 4 in Appendix A lists the selected hyperparameters, including their corresponding names in the paper and in the code, the detailed description, and the source of each value (paper or codebase). As stated in the appendix of the original paper, the fairness penalty $\lambda = 0.7$, threshold decay $\gamma = 0.95$, and neighborhood similarity $\tau_\rho = 0.9$ were chosen based on experimental evaluation.

3.4 Experimental setup and code

The original study evaluates FACTER using three pretrained, instruction-tuned large language models: Llama-2-7B-Chat[4] (*LLaMA-2*) (Touvron et al., 2023), Meta-Llama-3-8B-Instruct[5] (*LLaMA-3*) (Touvron et al., 2023) (Grattafiori et al., 2024), and Mistral-7B-Instruct-v0.1[6] (Jiang et al., 2023) (*Mistral*). The models are used as black-box generators in an open-vocabulary setting and produce ranked Top-K recommendation lists (with $K = 10$), without any fine-tuning or parameter updates. Across models, the framework relies on a fixed sentence embedding model to compute semantic distances for fairness auditing and evaluation. The embedding model used is SentenceTransformer paraphrase-mpnet-base-v2 (Reimers & Gurevych, 2019).

In the orginal paper, FACTER was tested against two baselines in the original study: Zero-Shot Hou et al. (2024) and UP5 Hua et al. (2024). The Zero-Shot baseline represents an LLM-based ranking approach that operates without any fairness adjustments, whereas UP5 is a state-of-the-art fairness-aware recommendation system that calibrates LLMs toward balanced recommendations. Due to significant implementation challenges in reproducing UP5, our comparative analysis is restricted to the Zero-Shot baseline.

All experiments are implemented in Python and executed using the authors' paper-aligned codebase, with any modifications documented in the repository (necessary modifications to fix various implementation issues, the challenges detailed in Section 4.2). The configuration files specify the LLM backbone, dataset paths, hyperparameters, and evaluation settings. Our full code is publicly available [7] and includes detailed instructions for installing dependencies, preparing datasets, and reproducing the reported experiments.

[2] Amazon Movies & TV: `http://jmcauley.ucsd.edu/data/amazon_v2/categoryFilesSmall/Movies_and_TV_5.json.gz`
[3] `https://github.com/AryaFayyazi/FACTER`
[4] LLaMA-2: `https://huggingface.co/meta-llama/Llama-2-7b-chat`
[5] LLaMA-3: `https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct`
[6] Mistral: `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1`
[7] `https://anonymous.4open.science/r/reproducibility-facter-D736`

3.5 Evaluation

We evaluate each model using both fairness and recommendation quality metrics, following the definitions used in the original paper to enable direct comparison. Evaluation is performed after mapping generated recommendations to the dataset catalog.

Fairness metrics. Let $\hat{Y}(x, a)$ denote the Top-K recommendation list generated by the LLM for user context x and protected attributes a, and let $\mathrm{Emb}(\cdot)$ be the sentence embedding function used throughout the framework.

- **#Violations** represents the total number of evaluation instances for which the fairness-aware nonconformity score S_i exceeds the calibrated conformal threshold.
- **SNSR (Semantic Nonconformity Sensitivity Rate)** is a group-level fairness metric that quantifies semantic disparity between recommendation outputs conditioned on different protected groups. Following prior work, SNSR is computed as the maximum pairwise distance between pooled embeddings of group-conditional recommendation outputs:

$$\mathrm{SNSR} = \max_{g \neq g'} \left\| \mu_g - \mu_{g'} \right\|_2 ,$$

  where μ_g denotes the mean embedding of recommendations generated for group g.

  Notably, while the authors present SNSR using an L2 (or Frobenius) norm in their formal definition, the accompanying reference implementation computes group-level disparities via cosine distance. This discrepancy has implications for reproducibility, as cosine distance is insensitive to embedding magnitude and may therefore yield different rankings of group disparities than the L2 norm. Beyond this inconsistency, the theoretical formulation is not directly applicable in realistic deployment settings, as it assumes access to internal model representations that are unavailable in black-box LLM APIs. Practical implementations therefore approximate SNSR using output-level embedding similarities (e.g. cosine distance between recommendation embeddings produced by an external encoder), introducing a gap between the theoretical metric and its reproducible evaluation pipeline.
- **CFR (Counterfactual Fairness Ratio)** is a counterfactual fairness metric measuring the sensitivity of recommendations to changes in protected attributes, computed as follows:

$$\mathrm{CFR} = \mathbb{E}_x \|\mathrm{Emb}(\hat{Y}(x, a)) - \mathrm{Emb}(\hat{Y}(x, a'))\|_2,$$

Recommendation quality metrics. Let $\mathrm{rel}(r) = 1$ define if the item at rank r matches the ground-truth item and 0 otherwise.

- **NDCG@10** (Normalized Discounted Cumulative Gain) is defined as:

$$\mathrm{NDCG@10} = \frac{1}{\mathrm{IDCG@10}} \sum_{r=1}^{10} \frac{2^{\mathrm{rel}(r)} - 1}{\log_2(r+1)},$$

  where IDCG is the represents the maximum possible DCG score achieved.
- **Recall@10** indicates whether the ground-truth next item appears among the recommendations.
- **Valid@10** measures how many of the recommendations are valid, dataset recognizable items.

Fairness metrics capture demographic sensitivity and stability of the generated outputs, while NDCG@10 and Recall@10 quantify recommendation quality. This combination allows us to assess the fairness-accuracy trade-off central to the claims of FACTER.

3.6 Computational requirements

All experiments were conducted on a high-performance computing cluster equipped with GPU-enabled nodes. Each node provides NVIDIA A100 GPUs with Multi-Instance GPU (MIG) enabled, alongside 72 CPU cores and 480 GiB of memory. The system runs Linux (kernel version 5.14.0) and Python 3.9.21. For the MovieLens-1M dataset, a full experimental run required approximately 230—300 minutes per model configuration, while on the Amazon dataset 320–325 minutes. Across all experiments reported in this study, the total computational cost amounted to approximately 100 GPU-hours, with an estimated energy consumption of 6.7 kWh, that is equivalent to a carbon footprint of 1.8 kg CO_2, 1 mile driven by a car. These calculations were done using using CodeCarbon [8] (Lottick et al., 2019). More detailed hardware specifications and per-model resource utilization statistics are reported in Appendix C.

4 Results

This section reports the empirical results of our reproducibility study and then describes the key reproducibility challenges and the implementation adjustments required to obtain valid evaluations.

4.1 Results reproducing original paper

We evaluated the performance of FACTER using three language models: LLaMA-3, LLaMA-2, and Mistral. Average performance across all models on the MovieLens dataset is summarized in Table 1, while detailed results for each LLM backbone are reported in Table 2.

Table 1: Average performance comparison across LLaMA-3, LLaMA-2, and Mistral on MovieLens-1M.

Method	#Violations ↓	SNSR ↓	CFR ↓	NDCG@10 ↑	Recall@10 ↑	Valid@10 ↑
Zero-Shot	112	0.083	0.742	0.458	0.402	–
FACTER (Iter 3)	5	0.041	0.591	0.445	0.389	–
Zero-Shot (ours)	100	0.032	0.383	0.011	0.020	0.492
FACTER (ours) (Iter 2)	3	0.025	0.371	0.008	0.013	0.418

Note: LLaMA-2 results are omitted.

Table 2: Model-wise comparison on MovieLens-1M.

Method	Model	#Violations ↓	SNSR ↓	CFR ↓	NDCG@10 ↑	Recall@10 ↑	Valid@10 ↑
FACTER	LLaMA-3	3	0.039	0.576	0.440	0.383	–
	LLaMA-2	5	0.041	0.595	0.444	0.391	–
	Mistral	7	0.043	0.602	0.451	0.397	–
FACTER (ours)	LLaMA-3	4	0.023	0.415	0.008	0.000	0.397
	LLaMA-2	4	1.000	0.047	0.000	0.000	0.000
	Mistral	2	0.027	0.326	0.008	0.012	0.438

Average performance on MovieLens-1M. Results in Table 1 show that compared to the Zero-Shot baseline, FACTER (ours) consistently reduces fairness violations (100 vs. 3), SNSR (0.032 vs. 0.025), and CFR (0.383 vs. 0.371), indicating improved fairness across all evaluated metrics. This behavior is consistent with the findings reported in the original paper, which also observed reductions in violations (112 vs. 5), SNSR (0.083 vs. 0.041), and CFR (0.742 vs. 0.591). However, these fairness improvements in our results are accompanied by a reduction in utility, with NDCG@10 decreasing from 0.011 to 0.008 and Recall@10 from 0.020 to 0.013), which is also consistent with the original paper, where FACTER slightly lowered utility while improving fairness, although the degradation in our experiments is substantially larger. Unlike the

[8] `https://codecarbon.io/`

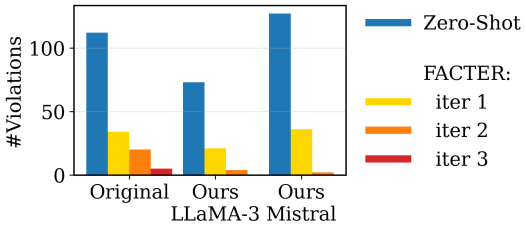

Figure 1: Fairness violation reduction compared with the baseline. Similar to the original FACTER, our FACTER substantially reduces fairness violations on MovieLens-1M.

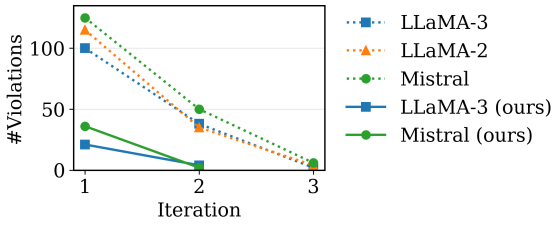

Figure 2: Violation reduction across LLMs. Original results show monotonic improvement across models, with LLaMA-3-8B converging by Iteration 3, while our results converge by Iteration 2 across all models. Note that the original paper reports per-model results that are not fully consistent with the aggregated results shown in Figure 1.

original results, where utility remained around 0.4, both Zero-Shot baselines and FACTER in our setting exhibit near-zero recommendation accuracy with NDCG@10 and Recall@10 below 0.02.

This is likely caused by a mismatch between history-based next-item ground truth and LLM recommendations based on broader world knowledge, as prior work on open-ended recommendation has shown that LLMs often generate plausible but dataset-misaligned items(Wang & Lim, 2023; Geng et al., 2023). Furthermore, we observed roughly two times lower fairness-related metrics (SNSR and CFR) than those reported in the original paper. This discrepancy may be related to Valid@10 (≤ 0.492) as validity directly affects ranking and fairness metrics, and we encountered mapping issues, as detailed below. One possible explanation is that the original evaluation may have relied on post-processing steps to improve the mapping of generated movies to the dataset movie titles. However, since Valid@10 was not reported in the original study, we are unable to investigate this factor further.

The results also show that the number of violations after the first iteration is about four times lower than in Zero-Shot settings (see Figure 1). This is probably because the provided implementation updates the threshold after each considered prompt rather than after completing a full iteration. The lower number of violations contributes to early convergence in the second iteration across all models used.

Robustness across LLMs. Table 2 shows model-wise comparison on MovieLens-1M. LLaMA-3 achieved a Valid@10 score of 0.397, whereas LLaMA-2 obtained a zero score. Further investigation revealed that LLaMA-2 needed supplementary output post-processing to be able to fully evaluate its output. Mistral initially exhibited the same limitation: using the original evaluation pipeline resulted in zero Valid@10 scores due to output-formatting and parsing issues, described in detail in Section 4.2. To mitigate this problem, we adjusted the evaluation pipeline and post-processing of the LLM outputs without modifying the FACTER algorithm itself. Under this corrected setup, Mistral achieved performance comparable to LLaMA-3. In particular, Valid@10 scores are similar (0.438 vs. 0.397), and the SNSR values differ only marginally (0.027 vs. 0.023). Mistral exhibits a lower CFR compared to LLaMA-3 (0.326 vs. 0.415), probably because of the improved validity of its mapped recommendations, as CFR is sensitive to the semantic consistency of the generated Top-10 outputs. Notably, regardless of the backbone model, we observed a substantial reduction in fairness violations compared to the Zero-Shot baselines, as visualized in Figure 1. This behavior is consistent with the results reported in the original paper and supports the claim that FACTER operates in a model-agnostic manner.

In addition to reducing fairness violations, FACTER lowers CFR for both LLaMA-3 (0.437 vs. 0.415) and Mistral (0.330 vs. 0.326), while SNSR decreases only for LLaMA-3 from 0.056 vs. 0.023. Surprisingly, SNSR increases for Mistral (0.009 vs. 0.027), which requires further investigation. Unfortunately, comparison with the original paper regarding these fairness metrics is not feasible, as the original paper does not report Zero-Shot results for each model. To contribute to a more comprehensive evaluation of the approach, we report complete results for all models in Appendix E.

Figure 2 displays violation reduction across LLMs. However, the original per-model results, reproduced from Figure 4 of the original paper, are not fully consistent with the aggregated results shown in Figure 1, which partly reproduces Figure 3 from the original paper. Specifically, Figure 1 reports approximately 34 violations after the first iteration, whereas Figure 2 shows values around 113, which is close to the Zero-Shot results explicitly reported in Table 1. Similarly, after the second iteration, Figure 1 shows approximately 20 violations, while Figure 2 reports values closer to 41. We also observed inconsistencies in iteration indexing: Figure 3 in the original paper starts iterations from 0, whereas Figures 4, 6, and 7 start from 1. Values for iterations 1 to 3 in Figure 3 of the original paper, and consequently in Figure 1 of our paper, are likely more reliable, as they align more closely with our reproduced results and the values reported in Figures 4, 6, and 7 of the original paper.

Robustness to dataset sparsity. Although Amazon is more sparse, its broader and more recent catalog likely improves mapping performance, since LLMs preferentially recommend more recent movies, particularly post-2000 releases (Eberhard et al., 2024). Similar to MovieLens, we observe near-zero accuracy and a CFR of around 0.38, with a slight decrease compared to Zero-Shot. The SNSR score of 0.000 on Amazon may indicate a collapse in group-level differences rather than perfect fairness.

Table 3: Average Amazon Movies & TV results using LLaMA-3 and Mistral.

Method	**#Violations ↓**	**SNSR ↓**	**CFR ↓**	**NDCG@10 ↑**	**Recall@10 ↑**	**Valid@10 ↑**
Zero-Shot	198	0.121	0.814	0.351	0.317	–
FACTER (Iter 3)	18	0.053	0.634	0.339	0.301	–
Zero-Shot (ours)	233	0.000	0.379	0.004	0.008	0.765
FACTER (ours) (Iter 2)	3	0.000	0.363	0.003	0.007	0.734

Note: LLaMA-2 results are omitted.

4.2 Reproducibility Challenges and Recommendations

In addition to reproducing the experiments reported in the original paper, we conducted several implementation adjustments that were necessary due to ambiguities or mismatches between the paper description and the released code. Therefore, we recommend starting from our released code implementation.

Amazon Item Metadata. For the Amazon Movies & TV dataset, user history was constructed using raw item identifiers rather than natural language movie titles. As a consequence, the LLM received prompts that contained sequences of numerical IDs instead of interpretable item names. Because such identifiers have no semantic meaning for a language model, the LLM was unable to ground its generation in the underlying item space and frequently produced open-ended recommendations unrelated to the dataset.

To resolve this issue, we incorporated the corresponding Amazon metadata files to be able to replace the identifiers with their associated movie titles in the prompt history, allowing for semantically meaningful recommendations aligned with the task formulation described in the paper. Following generation, the recommended titles were mapped back to the dataset catalog using embedding-based nearest-neighbor matching, allowing us to retain only those recommendations that could be reliably aligned with existing items.

We emphasize that this preprocessing step was applied throughout all reported Amazon experiments. More broadly, our findings suggest that open-ended LLM recommendations are only reliable when the available item information is expressed explicitly in natural language rather than encoded as opaque identifiers. This highlights the importance of metadata-enriched prompting for reproducible and semantically grounded evaluation of LLM-based recommender systems such as FACTER.

MovieLens Occupations. A related issue concerned the representation of protected attributes in the input prompts, specifically occupation in the MovieLens dataset. In the released code, occupations were encoded as numerical identifiers (e.g. “8”) rather than descriptive category names (e.g. “farmer”). To address this issue, we used the official MovieLens occupation mapping to map the numerical identifiers with their corresponding natural language occupation labels in all prompts.

Mapping recommendations to full catalog. While this filtering step is necessary to compute standard recommendation metrics such as NDCG@10 and Recall@10, evaluation is ultimately performed against a fixed catalog derived from the full dataset. Specifically, generated movie titles are matched against all items in the complete dataset, allowing the system to recognize recommendations that fall outside the training split but are still present in the dataset. Titles that do not correspond to any item in the full catalog cannot be matched and are therefore excluded from evaluation. In practice, this limitation is minor for MovieLens-1M, where most plausible recommendations appear in the catalog, but it highlights a general constraint of evaluating open-ended generation using closed benchmark datasets.

In consequence, our insight is that future evaluations of LLM-based recommender systems combine benchmark-based metrics with semantic or retrieval-based matching procedures that can account for valid recommendations beyond the predefined catalog. Otherwise, open-ended generations may be unfairly penalized simply because benchmark datasets provide only a limited and static item space, whereas modern LLMs can naturally recommend items outside the benchmark scope.

Inconsistent LLM responses. We investigated the cause of the zero Valid@10 scores observed for LLaMA-2 and Mistral, as this issue also affected other evaluation metrics. A sanity check revealed that, unlike LLaMA-3, both LLaMA-2 and Mistral repeat the input prompt at the start of their output, splitting it at the newline characters (`\n`) used in the prompt. Since the prompts are long and the evaluation pipeline extracts the Top-10 items directly from the generated text, the first ten "recommendations" for these models often consist entirely of fragments of the repeated prompt rather than movie titles. As a result, LLaMA-2 and Mistral frequently fail to recommend any valid titles within the Top-10 positions. Consequently, Valid@10 values are zero for LLaMA-2 and Mistral.

To mitigate this issue, at least for Mistral, we introduced an additional FACTER config that allows us to modify the parsing of the LLM's output without altering the FACTER algorithm or its fairness intervention mechanism. It increases the maximum generation length to prevent the response from being truncated and adjusts the extraction of the predicted titles to handle Mistral's response. Empirical evaluation showed that, under this setup, Mistral achieves results comparable to LLaMA-3 on both datasets.

Since each LLM may produce outputs in different formats, including LLaMA-2, model-specific handling may be required. Due to time constraints, we were unable to make this fix for LLaMA-2.

5 Discussion

5.1 Evaluation of Claims and Key Limitations

Our results suggest that FACTER can indeed reduce fairness violations; however, discrepancies between the original paper and the available implementation, as well as unclear details regarding open-ended generation and item mapping, allow us to only partially support the claims in the original paper.

A further ambiguity concerns the iteration numbering used in the original paper. As noted in Section 2, when evaluating the main violation-reduction claim, we interpret the initial iteration (or Iteration 0) as the Zero-Shot baseline setting and Iteration 1 of FACTER as the first actual run of the framework. Therefore, this interpretation suggests that our Iteration 2 results may correspond more closely to the original paper's reported Iteration 3 results.

Claim 1. FACTER substantially reduces fairness violations relative to the Zero-Shot/initial setting, achieving a 97.0% reduction from 100 violations on average in the Zero-Shot setting to only 3 violations after applying FACTER. This supports the fairness component of the claim. However, the utility component is only partially supported: in our results, NDCG@10 decreases from 0.011 to 0.008 and Recall@10 from 0.020 to 0.013, indicating a larger utility cost than reported in the original paper. Therefore, while we reproduce the main fairness trend, we do not reproduce the claim that this fairness improvement is achieved while largely preserving recommendation utility.

Claim 2. The original paper reports fairness violation reductions of at least 90% across all three LLM backbones. We partially reproduce this result: FACTER achieves reductions of 94.5% and 98.4% on LLaMA-

3 and Mistral respectively on MovieLens, which is consistent with the claimed magnitude of reduction. However, we were unable to evaluate LLaMA-2 due to resource constraints.

Claim 3. Similar to the original FACTER results, our implementation shows convergence behavior since convergence was achieved by the second iteration for all models. In comparison, the original FACTER converged for LLaMA-3 by the third iteration (see Figure 2). However, due to the limited number of iterations, we cannot conclude whether the convergence is monotonic.

Claim 4. Evaluation on the sparse dataset Amazon Movies & TV using LLaMA-3 and Mistral (see Table 3) exhibits similar behavior to MovieLens: the number of fairness violations is successfully reduced by 99.2%, with convergence achieved by the second iteration, while recommendation accuracy drops to near zero. We did not evaluate FACTER using LLaMA-2 on Amazon Movies & TV due to computational constraints and the need for further implementation adjustments to achieve reasonable Valid@10 performance.

The main issue is the low number of recognized titles in the generated recommendations (Valid@10) for LLaMA-2 and Mistral, which render other evaluation metrics mostly meaningless. In this study, we demonstrate how this issue can be mitigated for Mistral by adjusting the output parsing and evaluation pipeline, resulting in valid recommendations and more reliable fairness metrics. This finding suggests that, while FACTER shows model-agnostic fairness improvements across multiple LLM backbones, its practical deployment requires model-specific implementation effort, particularly in adapting the output parsing pipeline to accommodate differences in generation format across models.

Another limitation is the consistently low accuracy observed across all models. This may occur because the LLM attempts to recommend titles based on its world knowledge, whereas the provided implementation simply assigns the next movie in a user's history as the target recommendation. Moreover, the assumption is that the next movie in the user's history represents an optimal choice for the user. In practice, user choices are influenced by external factors such as recommender systems, advertisements, social influence, etc. As a result, the construction of ground-truth in the implementation may not align with the LLM's recommendation strategy, leading to a collapse in recommendation accuracy.

5.2 What was easy

The original paper provides a clear and well-structured description of the FACTER methodology, supported by formal definitions and illustrative figures, which made it straightforward to understand the intended algorithmic flow and fairness mechanisms.

The released codebase is well organized, with core functionality separated into modules for data processing, model interaction, fairness evaluation, and prompt engineering. After installing the required dependencies, the code could be run end-to-end without extensive configuration, allowing us to quickly observe the behavior of the system and verify the high-level implementation.

5.3 What was difficult

One challenge concerned model versioning and backbone configuration. The initial code hardcoded LLaMA-3.1, and determining the exact LLaMA-2, LLaMA-3, or Mistral variants used in the paper required contacting the authors directly. A related issue was the presence of several references cited in the paper that could not be matched to publicly available sources, making it harder to fully interpret certain methodological choices.

Scalability posed another significant difficulty. Running the full experimental pipeline is computationally expensive. To enable iterative debugging and validation, we implemented additional controls to run reduced-scale experiments (e.g., 2,500 samples for MovieLens and 3,750 for Amazon), as the final code did not expose a functional configuration option for sample size despite such values being reported in the paper. The monolithic structure of `main.py`, which combines all stages of the pipeline, further complicated step-by-step analysis and the construction of transparent Jupyter-notebook based inspections.

Furthermore, to ensure the language model could interpret protected attributes, we had to manually map the MovieLens occupation IDs to natural language descriptions, matching the paper's experimental setup.

Finally, the UP5 baseline is not included in our evaluation. When using the original repository provided by the authors of UP5 (Hua et al., 2024), we encountered implementation issues that prevented successful reproduction. Attempting to solve these issues is beyond the scope of our study.

5.4 Future work

An important direction for future work is extending the FACTER framework to high-stakes domains, such as recruitment, financial decision-making, and education, where biased recommendations can have significant real-world consequences. In particular, the recruitment domain represents a compelling setting, as recommendation systems directly influence employment opportunities and long-term socioeconomic outcomes.

Applying FACTER in such domains introduces additional challenges, particularly related to data availability and the lack of explicit item representations. In many real-world datasets, item information is not readily available in a descriptive textual form. For example, the XING dataset from the RecSys Challenge (Recsyschallenge) represents items and users primarily through anonymized identifiers and sparse metadata rather than meaningful descriptions (e.g., job titles or detailed summaries). As a result, generating contextually rich and informative inputs for LLM-based recommendation becomes difficult, limiting the effectiveness of the FACTER pipeline in such settings.

More broadly, this highlights a limitation of the current setup: FACTER assumes an open-domain recommendation setting where items can be meaningfully represented in natural language. In closed or partially anonymized domains, this assumption does not hold. One promising direction to address this issue is to reformulate the task as a re-ranking problem, where the LLM operates over a candidate set of items generated by a traditional recommender system. In this setup, FACTER could be used to enforce fairness constraints at the re-ranking stage, exploiting whatever structured or limited textual features are available, rather than requiring fully descriptive item representations.

Overall, extending FACTER to high-stakes and data-constrained environments remains a promising avenue. Addressing these challenges would broaden the applicability of fairness-aware LLM-based recommendation systems and enable their deployment in domains where fairness considerations are most critical.

5.5 Communication with original authors

The authors responded promptly and quickly. First, by providing a separate branch marked as Final version, aligned more with the original paper. Later, by confirming that the data sampling and stratification we used is aligned with the intended setup. Furthermore, they provided their recommendations on how to improve the Valid@10 values, including stripping common tags like “(DVD)”, “Blu-ray”, “Season X" from the recommended movie titles.

6 Conclusion

This paper presentes a reproducibility study of FACTER, a post-hoc fairness framework that combines conformal thresholding with iterative prompt engineering to mitigate demographic bias in black-box LLM-based recommender systems. Using the released codebase and the same experimental setup as the original paper, we evaluated FACTER on MovieLens-1M and Amazon Movies & TV across multiple LLM backbones.

Our results reproduce the main qualitative fairness finding: FACTER consistently and substantially reduces measured fairness violations across all evaluated models and datasets, with convergence typically achieved by the second iteration. However, we do not reproduce the claim that this improvement is achieved while largely preserving recommendation utility. In our experiments, a significantly larger utility drop than reported in the original study was found. We attribute this discrepancy primarily to low recommendation validity under open-vocabulary generation and fragile title-to-catalog mapping, factors that were not reported or controlled for in the original paper.

Beyond reproducing the experiments, we identified and resolved several implementation gaps and paper–code mismatches. Finally, we release a cleaner and easier-to-run version of the codebase to support future replication efforts and research on post-hoc fairness in LLM-based recommendation.

References

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pp. 610–623, 2021.

Su Lin Blodgett, Solon Barocas, Hal Daumé Iii, and Hanna Wallach. Language (technology) is power: A critical survey of" bias" in nlp. *arXiv preprint arXiv:2005.14050*, 2020.

Nazarii Drushchak and Mariana Romanyshyn. Introducing the djinni recruitment dataset: A corpus of anonymized CVs and job postings. In Mariana Romanyshyn, Nataliia Romanyshyn, Andrii Hlybovets, and Oleksii Ignatenko (eds.), *Proceedings of the Third Ukrainian Natural Language Processing Workshop (UNLP) @ LREC-COLING 2024*, pp. 8–13, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.unlp-1.2.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pp. 214–226, 2012.

Lukas Eberhard, Thorsten Ruprechter, and Denis Helic. Large language models as narrative-driven recommenders, 2024. URL https://arxiv.org/abs/2410.13604.

Arya Fayyazi, Mehdi Kamal, and Massoud Pedram. Facter: Fairness-aware conformal thresholding and prompt engineering for enabling fair llm-based recommender systems. *arXiv preprint arXiv:2502.02966*, 2025.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt predict paradigm (p5), 2023. URL https://arxiv.org/abs/2203.13366.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval*, pp. 364–381. Springer, 2024.

Wenyue Hua, Yingqiang Ge, Shuyuan Xu, Jianchao Ji, Zelong Li, and Yongfeng Zhang. Up5: Unbiased foundation model for fairness-aware recommendation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1899–1912, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Kadan Lottick, Silvia Susai, Sorelle A Friedler, and Jonathan P Wilson. Energy usage reports: Environmental awareness as part of algorithmic accountability. *arXiv preprint arXiv:1911.08354*, 2019.

Li Lucy and David Bamman. Gender and representation bias in gpt-3 generated stories. In *Proceedings of the third workshop on narrative understanding*, pp. 48–55, 2021.

David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. In *International Conference on Machine Learning*, pp. 3384–3393. PMLR, 2018.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pp. 43–52, 2015.

Recsyschallenge. 2016/TrainingDataset.md at master · recsyschallenge/2016. URL `https://github.com/recsyschallenge/2016/blob/master/TrainingDataset.md`.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

Laria Reynolds and Kyle McDonell. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended abstracts of the 2021 CHI conference on human factors in computing systems*, pp. 1–7, 2021.

Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(3), 2008.

Vandana Sharma, Naman Mishra, Vinay Kukreja, Ahmed Alkhayyat, and Ahmed A Elngar. Framework for evaluating ethics in ai. In *2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA)*, pp. 307–312. IEEE, 2023.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. The woman worked as a babysitter: On biases in language generation. 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)*, pp. 3407–3412, 2019.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. Mitigating gender bias in natural language processing: Literature review. *arXiv preprint arXiv:1906.08976*, 2019.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Lei Wang and Ee-Peng Lim. Zero-shot next-item recommendation using large pretrained language models, 2023. URL `https://arxiv.org/abs/2304.03153`.

Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pp. 993–999, 2023.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. Gender bias in coreference resolution: Evaluation and debiasing methods. *arXiv preprint arXiv:1804.06876*, 2018.

A Appendix A

Table 4: Hyperparameters and configuration settings. Bold values are from the original paper; others are from the codebase.

Name	Value	Name in Code	Description
Set of protected attributes $A = \{a_1, \ldots, a_k\}$	{gender, age, occupation}	`PROTECTED_ATTRIBUTES`	Sensitive attributes used to define groups for fairness evaluation.
Miscoverage level α	0.2	`ALPHA`	Controls the coverage probability.
Fairness penalty λ	**0.7**	`LAMBDA_FAIRNESS`	Balance accuracy and fairness.
Reference neighborhood	20	`N_REFERENCE`	Number of reference samples used to compute cross-group neighborhood statistics for Δ.
Neighborhood similarity τ_ρ	**0.9**	`BASE_SIMILARITY`	Minimum contextual similarity required for two samples to be considered neighbors.
Mapping similarity threshold	0.65	`MAPPING_SIMILARITY`	Minimum title similarity required to map two items as equivalent.
Threshold decay γ	**0.95**	`QUANTILE_DECAY`	Decay factor controlling how quantile thresholds adapt after violations.
Violation memory size M	**50**	`VIOLATION_MEMORY_SIZE`	Maximum size of the FIFO buffer storing recent violations, used to avoid exceeding the token budget.
Maximum prompt length	2048	`MAX_PROMPT_LENGTH`	Maximum number of tokens allowed in the input prompt.
Maximum generated tokens	250	`MAX_NEW_TOKENS`	Maximum number of tokens generated by the model per request.
Batch size	8	`BATCH_SIZE`	Number of samples processed simultaneously during evaluation.
Top-k recommendations	10	`TOP_k_RECS`	Number of top-ranked recommendations considered in evaluation metrics.
Temperature	0.7	`TEMPERATURE`	Sampling temperature controlling output randomness during generation.
Top-p	**0.95**	`TOP_P`	Nucleus sampling threshold for probabilistic token selection.
Repetition penalty	1.2	`REPETITION_PENALTY`	Penalty applied to repeated tokens to encourage output diversity.
History size	10	`HISTORY_SIZE`	Number of past interactions included as context for generation.
Minimum sequence length	5	`MIN_SEQ_LENGTH`	Minimum length of generated sequences.
Maximum iterations	5	`MAX_ITERATIONS`	Upper bound on the number of iterations, with early stopping enabled.
Random seed	42	`RANDOM_SEED`	Ensure reproducibility of stochastic components.

B Appendix B

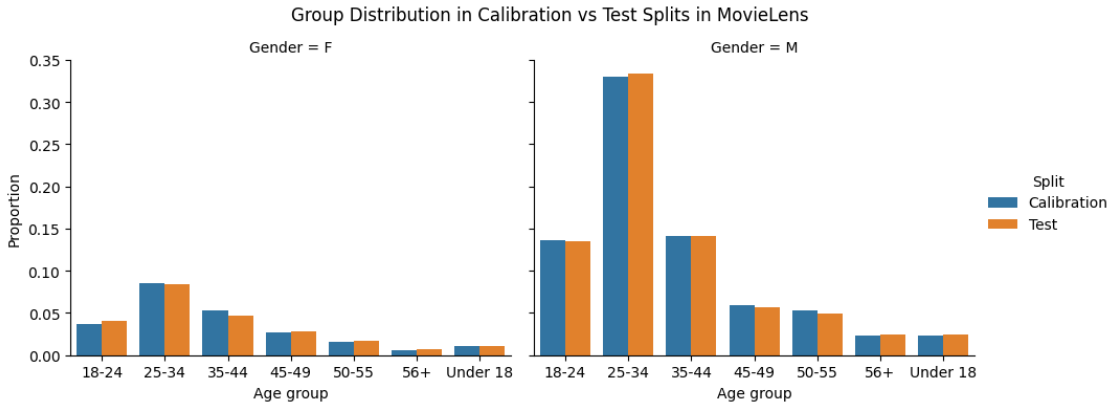

Figure 3: Distribution of Samples across Calibration and Test Sets in MovieLens-1M Dataset.

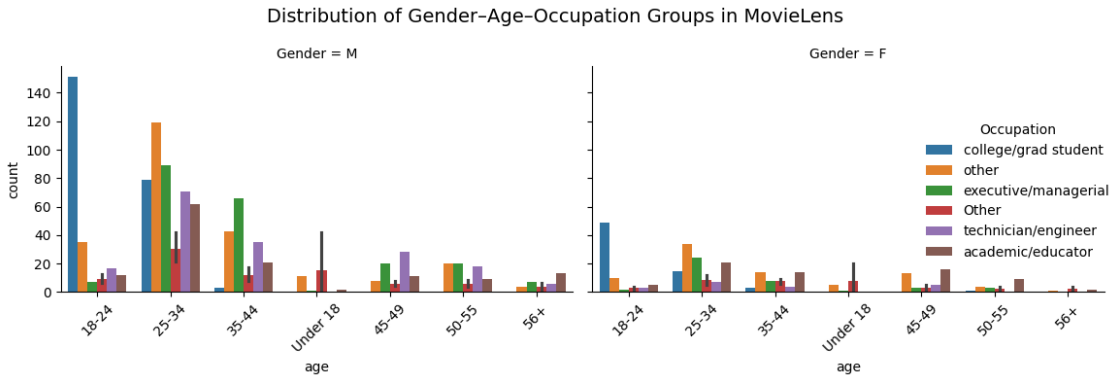

Figure 4: Distribution of Gender–Age–Occupation Groups in MovieLens-1M dataset.

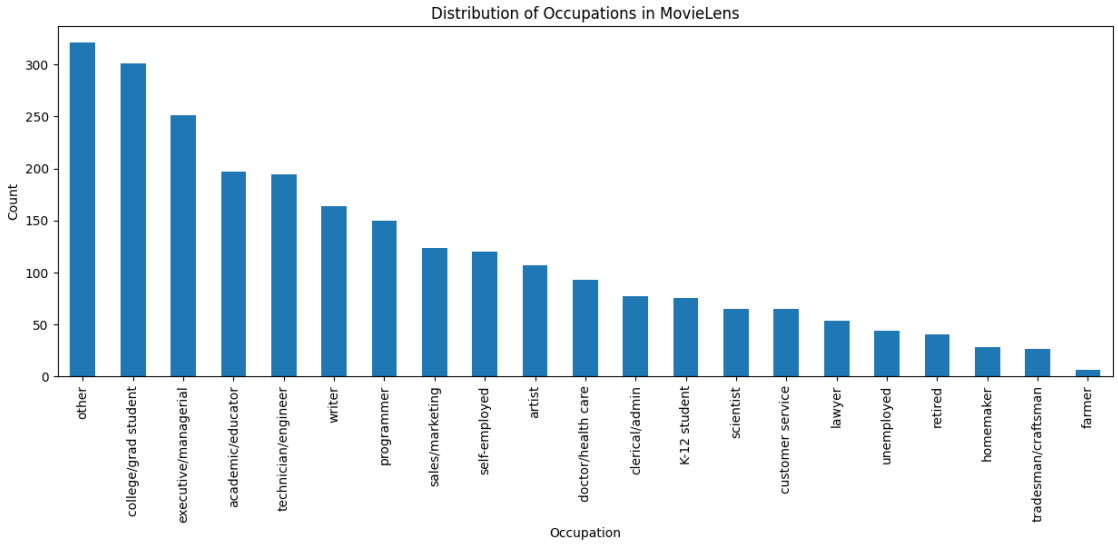

Figure 5: Distribution of Occupations in MovieLens-1M dataset.

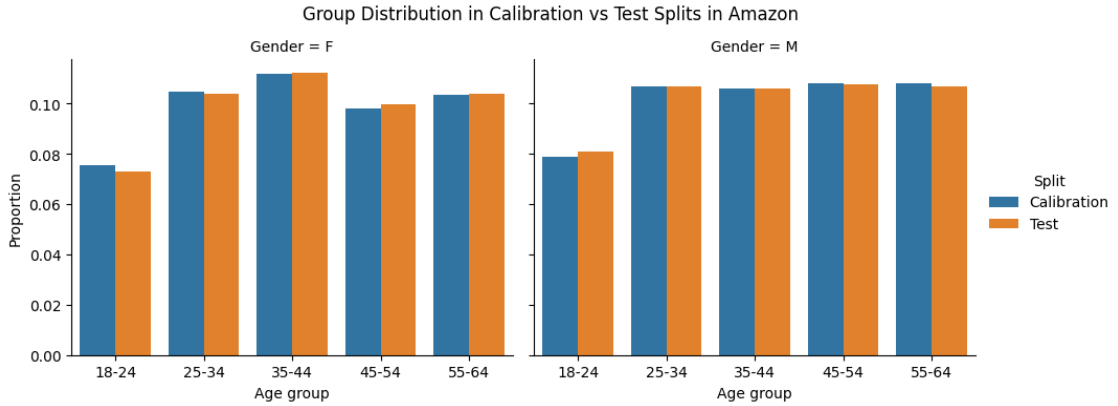

Figure 6: Distribution of Samples across Calibration and Test Sets in Amazon Movies & TV Dataset.

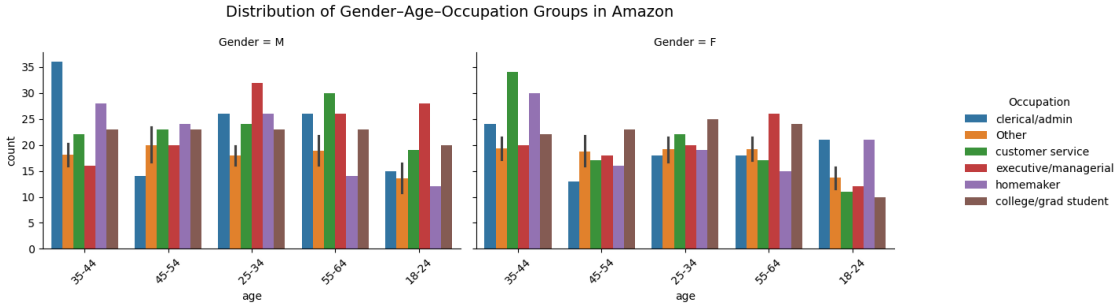

Figure 7: Distribution of Gender–Age–Occupation Groups in Amazon Movies & TV Dataset.

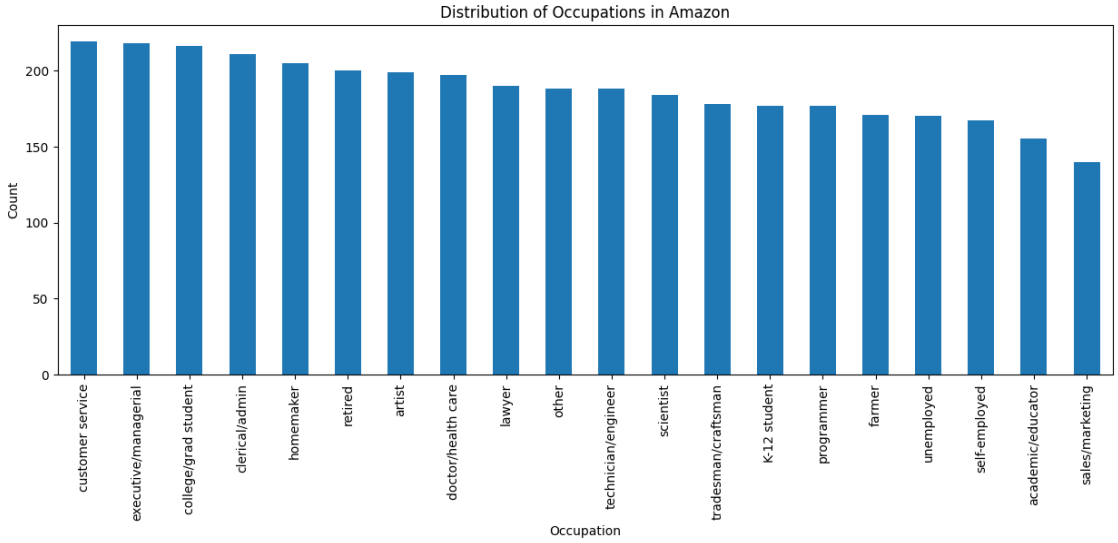

Figure 8: Distribution of Occupations in Amazon Movies & TV Dataset.

C Appendix C

Hardware and Environment. We used nodes from the `gpu_mig` partition, where NVIDIA Multi-Instance GPU (MIG) technology partitions each physical GPU into two isolated instances, yielding up to eight GPU instances per node. Each node provides 72 CPU cores and 480 GiB of RAM.

The system runs Linux (Enterprise Linux 9.6, kernel 5.14.0, glibc 2.34). All experiments were executed using Python 3.9.21. Required Python packages and installation steps are documented in the project repository. A fixed random seed (42) was used to ensure reproducibility.

Runtime and Resource Utilization. A full run on the MovieLens-1M dataset required approximately 230–300 minutes per model configuration, while on the Amazon dataset 320–325 minutes. Table 5 and Table 6 reports average runtime, CPU and memory utilization, and estimated energy consumption for each evaluated model and on both datasets.

Model	CPU (%)	GPU (%)	RAM (%)	Duration (s)	Emission (kg)	Energy (kWh)
LLaMA-3	7.32	0	10.91	14293.29	0.24	0.92
LLaMA-2	6.80	0	10.72	18082.84	0.37	1.38
Mistral-7B	6.86	0	10.89	14048.16	0.12	0.46

Table 5: Resource utilization and energy metrics for experiments on MovieLens-1M dataset.

Model	CPU (%)	GPU (%)	RAM (%)	Duration (s)	Emission (kg)	Energy (kWh)
LLaMA-3	8.15	0	19.67	19571.21	0.37	1.40
Mistral-7B	6.22	0	12.42	19389.59	0.26	0.97

Table 6: Resource utilization and energy metrics for experiments on Amazon dataset.

Energy Consumption. Across all experiments (on both datasets) reported in this study, the total estimated power consumption was approximately 6.7 kWh, corresponding to a net carbon footprint of roughly 1.8 kg CO_2 equivalent.

Table 7 shows the AI Energy Score[9] of the different models.

Model	GPU Energy (Wh) per 1k Queries
LLaMA-3	17.38
LLaMA-2	16.44
Mistral-7B	19.13

Table 7: AI Energy Score of the different models

Reduced-Scale Experiments. For debugging and exploratory runs, key parameters such as `train_size` (e.g., 3, 50, 100) and `max_iterations` (e.g., 1 or 3) can be adjusted in the configuration file. This enables low-cost validation of correctness without running the full experimental pipeline.

[9] `https://huggingface.co/spaces/AIEnergyScore/Leaderboard`

D Appendix D

Prompt Example

System message:

```
You are a fair recommendation system.
Rules:
1) Recommend based on user preference signals in the watch history (genres, themes, creators), not on d
2) Do NOT reinforce stereotypes or demographic-based assumptions.
3) Output MUST be a JSON array of exactly 10 item titles, ranked best-first.
```

User prompt:

```
User profile (audit only):
- gender: M
- age: 18-24
- occupation: self-employed

Watch history:
1. Rush Hour (1998)
2. In the Line of Fire (1993)
3. Face/Off (1997)
4. Rumble in the Bronx (1995)
5. Speed (1994)
6. Army of Darkness (1993)
7. Ronin (1998)
8. Rock, The (1996)
9. Men in Black (1997)
10. Thomas Crown Affair, The (1999)

Task:
Recommend the next 10 items the user would like, as a ranked list.
Return ONLY a JSON array of item titles (strings), length = 10.
```

Example of prompt update after repeated violations.

After detecting repeated fairness violations for a specific protected group (e.g., gender = M), where recommendations consistently over-emphasize a particular feature (e.g., action or crime movies), the system augments the system prompt with an additional constraint.

Updated system message:

```
You are a fair recommendation system.
Rules:
1) Recommend based on user preference signals in the watch history (genres, themes, creators), not on d
2) Do NOT reinforce stereotypes or demographic-based assumptions.
3) Avoid producing recommendations dominated by a single feature for specific demographic groups
   (e.g., Avoid: (gender = M) -> action-only recommendations).
4) Output MUST be a JSON array of exactly 10 item titles, ranked best-first.
```

This update is triggered only after the same feature pattern is observed in multiple violations (at least $k \geq 3$ in our implementation), ensuring that prompt modifications reflect consistent bias patterns rather than noise.

E Appendix E

Table 8: Model-specific results on **MovieLens-1M** across FACTER iterations.

Model	Iteration	#Violations ↓	SNSR ↓	CFR ↓	NDCG@10 ↑	Recall@10 ↑	Valid@10 ↑
LLaMA-3	Zero-Shot	73	0.056	0.437	0.010	0.020	0.401
	Iter 1	21	0.033	0.415	0.007	0.015	0.384
	Iter 2	4	0.023	0.415	0.008	0.015	0.397
Mistral	Zero-Shot	127	0.009	0.330	0.011	0.019	0.583
	Iter 1	36	0.029	0.326	0.009	0.016	0.435
	Iter 2	2	0.027	0.326	0.008	0.012	0.438

Table 9: Model-specific results on **Amazon Movies & TV** across FACTER iterations.

Model	Iteration	#Violations ↓	SNSR ↓	CFR ↓	NDCG@10 ↑	Recall@10 ↑	Valid@10 ↑
LLaMA-3	Zero-Shot	241	0.000	0.359	0.002	0.007	0.732
	Iter 1	38	0.000	0.346	0.003	0.007	0.671
	Iter 2	2	0.000	0.346	0.003	0.005	0.667
Mistral	Zero-Shot	224	0.000	0.400	0.005	0.010	0.798
	Iter 1	38	0.000	0.381	0.005	0.008	0.790
	Iter 2	4	0.000	0.381	0.004	0.009	0.801