Under Second-Round Review: *INFORMS Journal on Computing*

Echoes of Manipulation: The Reinforcing Effect of Preference Bias on External Perturbations in Recommender Systems

Liben Chen
University of Minnesota, chen7954@umn.edu

Meizi Zhou
Boston University, mzzhou@bu.edu

Jingjing Zhang
Indiana University, jjzhang@iu.edu

Gediminas Adomavicius
University of Minnesota, gedas@umn.edu

Abstract

Preference bias is a well-established phenomenon in recommender systems (RS), where the system-predicted rating that the user observes as part of the item recommendation (i.e., before consuming the item) ends up systematically impacting the self-reported preference rating that a user provides as feedback after consuming the item. This paper investigates how user preference bias amplifies and prolongs the adverse effects of *external perturbations* (e.g., shilling attacks) in RS. While prior research has treated preference bias and external manipulations as separate phenomena, we demonstrate through a series of agent-based simulation experiments that their interaction produces amplified, self-reinforcing adverse effects. Specifically, we simulate perturbation scenarios where a small subset of items is temporarily promoted through artificial rating inflation. We systematically vary user preference bias levels, the proportion of perturbed users and items, and the duration of perturbation across multiple experimental conditions. Our experiments reveal that even small-scale and transient perturbations can have lasting consequences on RS performance, especially when user preference bias is strong. Perturbed items continue to dominate recommendation lists long after the external shock ends, as biased user feedback reinforces and sustains their artificial prominence. This creates prolonged degradation of predictive accuracy, as well as reduced relevance and diversity of consumed items. Importantly, the adverse effects are consistent across different recommendation algorithms and rating datasets. We also find that unpopular and content-wise common items are especially susceptible to these adverse effects. We further demonstrate these adverse effects under different perturbation strategies. Finally, we formalize an optimization framework that characterizes how a strategic attacker can systematically exploit the interplay between external perturbations and users' preference bias to optimize the attack outcome under effort constraints. These findings point to the need for a bias-aware defense paradigm that jointly addresses adversarial attacks and biased user feedback.

Keywords: recommender systems, longitudinal dynamics, external perturbations, preference biases, agent-based simulation

1 Introduction

Recommender systems (RS) have become an integral component of modern online platforms, facilitating personalized content delivery in domains such as e-commerce, streaming services, and social media. They account for approximately 35% of Amazon's revenue, drive 90% to 95% of users' views on TikTok, and influence over 80% of viewing activity on Netflix (Gomez-Uribe Hunt 2015, McKinsey 2013, Hale 2021). Their success is largely attributed to the ability to enhance user experience by predicting individual preferences based on historical interactions. However, as these systems operate in increasingly complex and dynamic online environments, they face various external threats that compromise their reliability and competence (Gunes et al. 2014, Si Li 2020). One particularly pressing challenge is *external perturbations*, i.e., deliberate manipulations introduced by malicious actors to distort recommendation outcomes. The susceptibility of RS to such perturbations not only degrades their predictive performance but also raises serious concerns about trust, security, and fairness in digital platforms (Mobasher et al. 2007, Wang et al. 2024).

External perturbations encompass a broad range of adversarial influences on RS, with shilling (a.k.a. profile injection) attacks being among the most widely studied forms (Lam Riedl 2004), where adversaries inject fake user profiles into the system to either promote or demote specific items (Gunes et al. 2014). Common attack strategies, such as random attacks, average attacks, and bandwagon attacks, exploit vulnerabilities in collaborative filtering models by introducing deceptive rating patterns that skew recommendation outcomes (Mobasher et al. 2007, O'Mahony et al. 2004). These manipulations can have far-reaching effects, e.g., unfair exposure of items, degradation of recommendation quality, and erosion of user trust in the system (Burke et al. 2006).

Meanwhile, RS performance is shaped not only by external attacks but also by how users respond to system recommendations. A growing body of research has documented a phenomenon sometimes referred to as *preference bias*, where users' post-consumption ratings are influenced by the system's predicted ratings that the users had observed before item consumption rather than reflecting users' true preferences (e.g., Cosley et al. 2003, Adomavicius et al. 2013, 2022a). Prior research has demonstrated that users tend to align their ratings with the recommendations they receive, leading to distortions in the data used to train RS (Adomavicius et al. 2013, 2019), and such preference bias can impair RS prediction accuracy and reduce consumption relevance and diversity over time (Zhou et al. 2024).

Despite extensive research on the detection and mitigation of external attacks and on the effects of preference bias, the dynamic interplay between these two factors remains underexplored. It is not well understood whether and how preference bias might interact with external perturbations through the RS-user feedback loop, potentially amplifying and prolonging the negative consequences of attacks. In this paper, we explore the possibility that these two forces—external perturbations and preference bias—may interact in a way that produces emergent, compounded vulnerabilities.

We posit that preference bias may function as an internal engine, transforming a short-term external shock into a self-reinforcing feedback loop of performance degradation. In the presence of preference bias, an attacker may need only a small perturbation to initiate a cycle: distorted recommendations lead to biased feedback, biased feedback becomes part of the training data used to retrain the future iterations of the recommendation model, which further leads to skewed future recommendations and additional distorted

user feedback, etc., thus perpetuating the attack long after it ends. This represents a critical and previously uncharacterized vulnerability in the design and implementation of RS.

To examine this, we employ an agent-based simulation framework to model the longitudinal dynamics of RS under varying levels of both external perturbations and user preference bias. Our results demonstrate that preference bias substantially magnifies and prolongs the negative impacts of external attacks, degrading predictive accuracy, item relevance, and consumption diversity over time. Notably, even small-scale, short-lived attacks that target merely 20% of users can produce long-lasting effects when combined with preference bias, and these effects are facilitated disproportionately by *less popular* and/or *more content-wise common* items.

This work is, to our knowledge, the first to systematically analyze how internal behavioral biases may amplify and prolong the adversarial effects of short-term external attacks. We move beyond static models of RS vulnerability to reveal dynamic feedback loops that can sustain recommendation manipulation over time. This study makes both theoretical and practical contributions. Theoretically, we provide the first systematic analysis of how endogenous user preference biases and exogenous attacks interact within RS feedback loops. Our work thus contributes to a comprehensive understanding of the long-term vulnerabilities of RS. Practically, we formalize this vulnerability as a strategic-attacker-oriented optimization problem, showing that an adversary aware of the bias-driven amplification mechanism can substantially expand damage at the same effort budget; the resulting threat model exposes blind spots in existing perturbation mitigation strategies, which often overlook behavioral feedback mechanisms, and motivates a bias-aware mitigation paradigm. In summary, this work reframes preference bias not just as a behavioral phenomenon, but as a latent security vulnerability that can be exploited to amplify and perpetuate the effect of RS attacks.

2 Related Work and Conceptual Framing

2.1 External Perturbations on RS

Recommender systems (RS) have been increasingly deployed in diverse and complex online environments, where protecting their security and reliability from external perturbations has become a growing challenge. One major type of external perturbation is *shilling attacks*, also known as profile injection attacks, where adversaries manipulate recommendations by injecting fake user profiles. These attacks typically serve two main purposes: push attacks, which aim to artificially promote a target item through inflated ratings; and nuke attacks, which aim to demote an item through disproportionately low ratings (Gunes et al. 2014, Si Li 2020, Deldjoo et al. 2021). Such manipulations degrade recommendation quality, distort item rankings, and reduce user trust (Lam Riedl 2004, Tang et al. 2020, Zhang et al. 2021, Li et al. 2022, Nguyen et al. 2024).

Shilling attacks employ different strategies to exploit RS vulnerabilities. A common example is the *random attack*, where attackers generate fake user profiles with ratings randomly distributed across items, except for the target item, which is rated extremely high or low. Although relatively easy to detect due to its irregular rating patterns, this attack can harm RS performance by introducing noise (O'Mahony et al. 2004). A more sophisticated approach is the *average attack*, where fake profiles mimic the overall rating distribution of real users, making them harder to detect (Mobasher et al. 2007). This method is effective

against collaborative filtering models that rely on similarity measures. Another widely studied strategy is the *bandwagon attack*, which takes advantage of the popularity bias in RS by injecting fake profiles that rate both popular items and the target item highly (Si Li 2020). Because popular items are frequently rated by real users, this tactic allows fake profiles to blend in more easily. By associating the target item with frequently rated popular items, the attack increases the chances of recommending the target item. These attacks are symmetric in that they can be constructed to either promote or demote targeted items, and the key insights from this study remain valid for either attack direction.

Understanding the impact of external perturbations on RS is crucial for maintaining RS reliability, effectiveness, and security. These attacks not only reduce predictive accuracy but also distort exposure and market dynamics (Mobasher et al. 2007, Nguyen et al. 2024). In domains such as e-commerce and streaming, such attacks can lead to financial losses, unfair exposure, and consumption of low-quality content (Burke et al. 2006, Di Noia et al. 2020, Cao et al. 2020). In sensitive areas such as healthcare, the consequences can extend beyond commercial loss to ethical and social concerns (Newaz et al. 2020). A deeper understanding of such external perturbations can enable researchers to develop more effective mitigation techniques (Wang et al. 2024, Ge et al. 2024). In particular, it has motivated the stream of adversarially robust RS literature that emphasizes reliability under poisoning and manipulation (Shehmir Kashef 2026, Zhang et al. 2025, Fan et al. 2023).

2.2 Behavioral Biases and Recommender Systems

A large body of research has examined various forms of user behavioral biases that emerge from both pre- and post-consumption interactions between users and RS. These biases can be broadly categorized into *selection biases* and *preference biases*, each influencing different stages of user-system interaction.

Selection biases primarily stem from users' pre-consumption behaviors, reflecting how item *choices* are shaped by the recommendations they receive (Ovaisi et al. 2020, Wang et al. 2021, Chen et al. 2023). *Popularity bias* leads users to engage more frequently with popular items, making them more likely to be recommended and consumed (Abdollahpouri 2019, Abdollahpouri et al. 2021, Klimashevskaia et al. 2024, Steck 2011, Zhu et al. 2021). *Position bias* describes users' tendency to interact with items based on their position in a recommendation list, irrespective of their actual relevance (Hofmann et al. 2014, Collins et al. 2018). Additionally, *exposure bias* arises when users are selectively shown a narrow subset of items while being underexposed to the rest (Liu et al. 2020, Chen et al. 2023). These selection biases contribute to phenomena such as *filter bubbles* and *echo chambers*, raising concerns about fairness and representativeness in recommendation outcomes (Abdollahpouri et al. 2019, 2020, Ge et al. 2020, Sukiennik et al. 2024).

In contrast, preference biases manifest themselves via users' *post-consumption* behaviors. One common example is the phenomenon where users' post-consumption *ratings* are influenced by the initial recommendations the users received at the selection stage, rather than by the users' independent evaluation of the consumed items (Cosley et al. 2003, Adomavicius et al. 2013, 2018). Preference biases have been documented across various domains (e.g., movies, music, television, and jokes), with different recommendation methods (e.g., personalized vs. non-personalized systems), and under various rating presentation formats (e.g., numerical vs. graphical vs. binary displays) (Adomavicius et al. 2013, 2019). Post-consumption ratings are typically

assumed to reflect users' true preferences, as they are routinely used as ground-truth data for training and evaluating RS algorithms. However, if these ratings are systematically distorted by preference biases, they may lead to suboptimal model development. In particular, Zhou et al. (2024) have shown that preference biases significantly impair the system's prediction performance (i.e., prediction accuracy) as well as users' consumption outcomes (i.e., consumption relevance and diversity) over time.

Taken together, such evidence suggests the need for debiasing and motivates the stream of literature on causal recommender systems, where logged feedback must be interpreted under exposure, confounding, and recommendation-induced expectations rather than as unbiased revealed preferences (Chaney et al. 2018). Recent causal recommender-system research has begun to instantiate this agenda with concrete estimators and architectures, including model-agnostic causal learning (Zhang et al. 2026) and causality-aware sequential preference modeling (Wang et al. 2026).

2.3 External Perturbations and Preference Bias: A Self-Reinforcing Cycle

Although both external perturbations and preference biases are well-documented phenomena, prior research has investigated them as fundamentally separate challenges: the former as an exogenous security issue, the latter as an endogenous behavioral bias. In other words, the possibility that, in combination, they could create a longitudinal performance degradation loop in RS has not been systematically explored. This oversight may arise from the implicit assumption that these two vulnerabilities, one exogenous and one endogenous, operate in separate spheres. However, in feedback-driven systems where user input is continually recycled into model training, the separation between the two may disappear; in fact, preference bias may serve as a powerful amplifier, enabling even small or transient external perturbation attacks to have long-lasting and systemic consequences.

At its core, the existence of preference bias means that users' post-consumption ratings are not fully independent of the system (i.e., are not driven solely by users' own inherent tastes and preferences) but are instead partially aligned with the system-generated predictions (that were observed by the users). This creates a fundamental vulnerability within the RS feedback loop. When the system has been externally perturbed–whether through malicious external shilling attacks or just through some inherent predictive inaccuracies of the recommendation model–users are more likely to receive and consume manipulated recommendations. These distorted recommendations, shown during the attack window, are then subject to preference-biased feedback. That feedback, in turn, re-enters the system as training data, contaminating the model. As the system retrains on biased inputs, the distortion can persist or even intensify.

Specifically, this iterative mechanism constitutes a self-reinforcing cycle, where the effect initiated by an external actor is perpetuated by users' behavioral bias. The cycle begins when a temporary perturbation alters the system-predicted ratings, exposing users to distorted recommendations. User feedback, influenced by preference bias, tends to align with the observed (distorted) ratings rather than reflecting solely the users' true preferences. The distorted consumption and biased user-specified ratings enter the training data, which is likely to affect the (retrained) model, and thereby affect the subsequent recommendations as well. These recommendations are then shown to users as part of future user-system interactions, thus continuing the self-reinforcing cycle.

As this loop continues (i.e., even after the external perturbation ceases), a one-time perturbation could become entrenched due to the system's internal logic. Rather than diminishing naturally over time, the negative performance effects linger (and in some cases could potentially even intensify) because they are being reinforced by users' preference biases. That is, this feedback loop can transform a temporary exogenous shock into a more sustained performance reduction.

We contend that this self-reinforcing cycle represents a *latent security vulnerability* of RS, where the external attacker's effort is magnified by the inherent processes of RS itself, as well as by users' inherent behavioral biases. In this context, preference bias is no longer just a measurement artifact – it represents an inherent "engine" that can make the perturbation attacks more pervasive and persistent. What makes preference bias an especially *insidious* amplifier among the many factors that moderate external-attack impact is its *stealth*: the resulting downstream contamination is generated by *honest* users behaving normally under a well-documented behavioral phenomenon (Tversky Kahneman 1974, Chapman Johnson 2002, Epley Gilovich 2010, Adomavicius et al. 2013) rather than by detectable adversarial attacks, so the biased post-consumption ratings carry no overt signature of manipulation. As a consequence, this amplification channel is structurally invisible to existing intrusion-side defenses such as sybil-detection methods, account-layer identity-verification controls, and behavioral-fingerprinting techniques (Si Li 2020, Deldjoo et al. 2021), and an external perturbation can therefore continue to propagate through the user–RS feedback loop long after the adversarial attacks have been detected and removed. Understanding and modeling the interplay between external perturbations and preference bias is important for developing *robust* and *resilient* RS, i.e., RS that are not only robust to initial perturbations, but also resilient to the longitudinal effects propagated through user feedback. This work does this by extensively and systematically quantifying and characterizing the downstream risks of preference bias in the context of adversarial manipulation.

3 Simulation Framework

We employ an agent-based simulation methodology to investigate the longitudinal impact of external perturbations on RS under preference biases. This simulation-based approach offers a robust and flexible environment for systematic exploration, enabling targeted manipulations and precise isolation of various factors, including platform type, specifications of the external perturbations, user population characteristics, item population attributes, recommendation algorithms, users' susceptibility to bias, and consumption frequency. Furthermore, the framework facilitates the analysis of intermediate processes, thereby providing insights into the underlying mechanisms through which external perturbations, preference bias, and their interactions influence the performance of RS.

The simulation approach offers several advantages over alternative methodologies. Field experiments struggle to control variables like perturbation magnitude and preference bias, and the adverse effects of external perturbation and preference bias may result in significant financial losses for companies, making such experiments ethically and practically problematic. Compared to laboratory experiments, simulation enables large-scale user interactions with RS over hundreds of iterations, capturing more extensive behavioral dynamics. Compared to analytical modeling, which can provide strong theoretical insights but often requires

Table 1: Outline of the Simulation Steps

Initialization
Step 0: System Initialization
Initialize $UserTastes$ for all users based on real-world rating data.
Initialize $ItemContent$ for all items based on real-world rating data.
Initialize pre-existing ratings R^0.
Main Simulation (perform Steps 1 and 2 at t = 0, 1, 2, 3, . . .)
Step 1: Recommendations under Perturbation
Use all available rating data at current time t: R^{t-1}, to train the RS algorithm.
Calculate item predictions $P^t = PredictRatings(R^{t-1})$.
Externally perturb the predicted ratings as $\tilde{P}^t_{ui} = P^t_{ui} + \pi_{uit}$, where
$\pi_{uit} = \begin{cases} \delta_{uit}, & \text{if } u \in \mathcal{U}_\pi,\ i \in \mathcal{I}_\pi,\ t \in \mathcal{T}_\pi \\ 0, & \text{otherwise} \end{cases}$, to obtain perturbed predictions $\tilde{P}^t$.
Generate the recommendation list for each user u, i.e., $L^t_u = GenerateRecs(\tilde{P}^t_u)$.
Step 2: Main Interaction/Consumption/Feedback Process
For each user u who consumes at time t:
u selects item i from recommendations L^t_u in accordance to $UserConsStrat_u$.
u consumes item i and submits her preference rating S_{ui}.
Update rating data to include the new rating, i.e., $R^t = R^{t-1} \cup S_{ui}$.
Update next consumption time for u, i.e., $UserNextCons_u = UserNextCons_u + UserConsPeriod_u$.
Evaluate recommendation performance.

explicit and simplifying mathematical assumptions (e.g., about the RS algorithm learning process), simulation can directly incorporate (and analyze the outcomes of) a wide range of the actual, highly complex, real-world RS techniques, such as deep-learning-based RS algorithms.[1] Still, the usefulness of simulation depends on accurately reflecting real-world conditions. Accordingly, we model external perturbations as inflated RS-predicted ratings, and preference bias as distorted user-reported post-consumption ratings. We build upon established agent-based simulation frameworks from prior literature (Zhang et al. 2020, Zhou et al. 2024) and extend them by modeling external perturbations and exploring different perturbation settings to assess their impact on long-term system dynamics under various preference bias conditions.

3.1 General Procedure

Table 1 presents an overview of the simulation procedure. Within each discrete time period, a subset of users engages with the platform, upon which the RS generates and delivers personalized recommendations. Users may then select items for consumption and provide feedback ratings. The system subsequently integrates all users' feedback ratings to update its model and refine recommendations for the next time period.

[1] In this view, the two approaches (analytical modeling and simulation) can be complementary. Thus, in addition to the simulation-based analysis that constitutes the bulk of this paper (and as part of extensive robustness checks, as enumerated in Section 7), we also develop a tractable closed-form analytical model (using some simplifying assumptions about RS algorithm behavior) in Appendix A to provide additional theoretical insights about the interplay between external perturbations and preference bias.

Simulation Initialization We first initialize the three core components—item population, user population, and recommendation engine—according to the specified distributions (Step 0 in Table 1). We model item and user populations with latent representations. Specifically, we use a vector of K numeric latent features, i.e., $ItemContent_i = (q_1, \ldots, q_K)$, to represent the content of the item i. Likewise, we represent user preferences for items using a vector of K numeric latent features, i.e., $UserTaste_u = (p_1, \ldots, p_K)$. We keep user and item sets static throughout the simulation period to avoid additional complexity from new user or item arrivals.

To ensure realistic latent factors, we fit $ItemContent$ and $UserTaste$ using real-world rating data with the FunkSVD algorithm (Funk 2006). This method factorizes the rating matrix into low-rank user and item embeddings—$UserTaste_u$ and $ItemContent_i$—by minimizing a regularized squared difference between predicted and observed ratings. The optimal parameters for FunkSVD are determined through standard practices such as grid search and cross-validation. At time $t = 0$, the recommendation engine is initialized with all preexisting ratings R^0 from the real-world data.

Ground-Truth Preference Ratings The derived latent vectors are used to instantiate users' "ground-truth" preference ratings over items. Empirical evidence indicates that users exhibit idiosyncratic noise and temporal inconsistency in ratings (Amatriain et al. 2009a). Following this literature, we model each user's true preference ratings as a normally distributed value centered on the "ground-truth" preference with a standard deviation of 1.0 on a 1–5 scale. Formally, the preference rating r_{ui} for a user u on item i is computed as:

$$r_{ui} = ItemContent_i \cdot UserTaste_u + \varepsilon_{ui}, \quad \varepsilon_{ui} \sim \mathcal{N}(0, 1) \tag{1}$$

Rather than treating real-world ratings as the "ground truth" preference ratings, we simulate them via Equation 1, since real datasets cover only a small subset of user-item pairs. Our simulation assigns "ground-truth" values for every user–item interaction to enable feedback for any consumed item. These ground-truth ratings are not directly observed by the recommendation algorithms; they are only used later to determine user-submitted ratings after consumption. Consistent with real-world practice, the RS is trained on submitted (rather than ground-truth) ratings, which may systematically differ from the underlying true ratings due to factors such as external perturbations and user preference biases.

User Consumption Choices At each subsequent period $t \geq 1$, RS predicts every user's preference ratings for unconsumed items using a recommendation algorithm. For user u, the system presents a ranked list L_u^t based on the displayed ratings $\tilde{P}_u^t$. Users' selection probabilities are rank-dependent, with higher ranks more likely to be chosen and consumed. Consistent with prior work (Chen et al. 2023, Zhang et al. 2020, Collins et al. 2018, Ursu 2018, Hofmann et al. 2014, Carare 2012, Joachims et al. 2007, 2005, Granka et al. 2004), we assume an exponentially decaying consumption probability by rank; specifically, the probability of consuming the item in position i is:

$$prob_i = (\alpha - 1) \cdot \alpha^{-i} \tag{2}$$

Equation 2 uses α to control how strongly users rely on recommendations. A larger α increases the likelihood of selecting higher-ranked items. Throughout the main experiments, we set $\alpha = 2$, under which items at the rank 1, 2, 3 have consumption probabilities of 50%, 25%, 12.5%, respectively, and the top five items together account for roughly 97% of the consumption probability mass.

To model user-specific consumption frequency, we define $UserConsPeriod_u$ as the average timespan

between two consecutive consumptions for user u (e.g., a value of 2 means one item every two periods on average). These timespans are calibrated from real-world data and mapped to the range $\{1, \ldots, 5\}$. During initialization, $UserNextCons_u^0 \in \{1, \ldots, 5\}$ is sampled at random; in later periods, the intervals are drawn from a normal distribution with mean $UserConsPeriod_u$.

External Perturbation We model external perturbation on RS as an artificial inflation of the predicted ratings. The perturbations are specified by four key factors.

- *Perturbed Items:* $\mathcal{I}_\pi \subseteq \mathcal{I}$ denotes the subset of items that are affected by external perturbations. The composition of $\mathcal{I}_\pi$ is determined by the *perturbed item ratio* (e.g., 1% or 0.1% in our experiments) and the *item type* (e.g., random, niche-content, or popular items).
- *Perturbed Users:* $\mathcal{U}_\pi \subseteq \mathcal{U}$ denotes the subset of users influenced by perturbation, determined by the *perturbed user ratio* (e.g., 0%, 5%, 20%, 100%) and *user type* (e.g., random or high-consumption-frequency users).
- *Perturbation Period:* $\mathcal{T}_\pi \subseteq \mathcal{T}$ denotes the time periods during which the external perturbation is active (e.g., the first 5, 10, or 15 timesteps in our experiments).
- *Perturbation Magnitude:* $\pi_{uit} > 0$, if $u \in \mathcal{U}_\pi,\ i \in \mathcal{I}_\pi,\ t \in \mathcal{T}_\pi$, otherwise $\pi_{uit} = 0$. The magnitude may represent either a 1-star rating increase or the minimum increase that can promote an item into the top of the recommendation list.

User Post-Consumption Feedback Once an item is consumed, the user submits a feedback rating reflecting the item's perceived relevance. For each user-item pair (u, i), the submitted rating S_{ui} depends on user's true preference r_{ui} and on the preference bias $bias$ induced by the system's displayed (and possibly perturbed) rating $\tilde{P}_{ui}^t$:

$$S_{ui} = r_{ui} + (\tilde{P}_{ui}^t - r_{ui}) \cdot bias \tag{3}$$

The $bias$ parameter ranges from 0 (where the submitted rating equals the user's true preference) to 1 (where the submitted rating fully mirrors the system-predicted rating). Because prior research indicates a continuous linear relationship between displayed and submitted ratings through a rigorous user study (Adomavicius et al. 2013, p. 972, Fig. 7), Equation 3 employs a linear formulation that pulls the submitted rating toward the displayed value in proportion to the strength of $bias$.[2]

Simulation Assumptions The simulation abstracts the real-world RS and, by construction, omits some real-life complexities. Specifically, we assume that: (1) each user-item pair involves no repeat consumption; (2) monetary factors do not affect user choice; (3) items have no inventory limits and can be consumed simultaneously by multiple users (as with digital goods); (4) recommendations are generated by a single, uniform system; and (5) the study isolates preference bias in canonical recommendation algorithms, excluding other potential influences (such as aggregate ratings, social networks, expert advice, and prior knowledge).

[2]This is theoretically grounded in the decision-making theory, specifically, the anchoring-and-adjustment heuristic (Tversky Kahneman 1974, Chapman Johnson 2002, Epley Gilovich 2010); and Adomavicius et al. (2013) operationalizes it in the RS context.

Performance Evaluation Recommendation quality is evaluated on three axes: *prediction accuracy*, *consumption relevance*, and *consumption diversity*.

For *prediction accuracy*, we measure how accurately RS predicts user preferences for unconsumed items using the standard $RMSE$ metric. As users consume items over time, the pool of unconsumed pairs contracts; to ensure comparability, we evaluate against the full set of unknown ratings defined at $t = 0$. At time t, $RMSE$ is computed from P^t versus R:

$$RMSE_t = \sqrt{\frac{1}{|T|} \sum_{(u,i) \in T} (P^t_{ui} - r_{ui})^2} \tag{4}$$

Second, *consumption relevance* provides user-centered assessment of RS performance that captures the quality of users' actual consumption experiences. It is calculated as the mean of users' true preference ratings for consumed items at time t:

$$Relevance_t = \frac{1}{|C_t|} \sum_{(u,i) \in C_t} r_{ui} \tag{5}$$

Third, *consumption diversity* measures RS ability to help users discover a broad, diverse set of relevant items. We focus on the population-level consumption diversity, a.k.a. *aggregate* diversity (Adomavicius Kwon 2014), which is measured as the cumulative number of unique items consumed by all users up to (and including) time t:

$$Diversity_t = |Q_1 \cup Q_2 \cup ... \cup Q_t|, \quad Q_t = \{i | (u,i) \in C_t\} \tag{6}$$

3.2 Application Settings

We examine two application settings by initializing the simulation with public datasets from distinct domains. Specifically, we use (i) a subset of the Netflix 100M dataset (Bennett Lanning 2007), consisting 3,000 randomly chosen movies and 3,000 randomly sampled users who rated at least one of those movies, and (ii) a subset of the Yelp ratings dataset, consisting of 2,000 randomly selected restaurants and 4,000 randomly sampled users. Ratings are integers from 1 to 5 (1 = least preferred, 5 = most preferred). Table 2 summarizes these datasets. User and item embeddings are represented by 20 latent features in the Netflix-based setting and 150 in the Yelp-based setting.[3]

Each discrete simulation timestep $t = 1, 2, \ldots$ represents one *recommender-engine update cycle*: on each cycle, the platform may refresh recommendations after incorporating new ratings, and users consume according to the interaction schedule described in Table 1.[4] User-specific consumption intensity is seeded from the empirical rating histories—specifically, we calibrate $UserConsPeriod_u$ from observed spacing between each user's rating events in the raw logs, initialize $UserNextCons^0_u$, so heterogeneity in real-world activity rates informs heterogeneity in simulated daily consumption. Under this mapping, a five-period perturbation window corresponds to five consecutive days of inflated predictions, and a T-period horizon corresponds to roughly T days of simulated operation.

[3]For FunkSVD initialization, we grid-search the latent dimension over $K \in \{10, 15, 20, 25, 50, 100, 150, 200\}$ and select K by cross-validation on the observed rating matrix of each seeding subsample (holding out ratings for validation), minimizing validation prediction error. The resulting choices $K = 20$ (Netflix subsample) and $K = 150$ (Yelp subsample) follow from this tuning.

[4]In our simulation setup, the platform retrains the system daily.

We compare several widely used recommendation algorithms, including an item-based nearest-neighbor method (ItemKNN) (Sarwar et al. 2001), a matrix factorization approach based on SVD (Koren 2008), a reinforcement learning model (LinUCB) (Li et al. 2010), and a graph-based contrastive learning model (MHCL) (Li et al. 2025). To ensure robustness, all simulations were conducted 10 times, with the reported results representing the averaged outcomes. Additionally, Appendix B provides details on the final hyperparameter configurations and grid search procedures employed for all models.

4 Experiment 1: How Preference Bias Amplifies and Perpetuates Perturbation

Experiment Setting In Experiment 1, we evaluate how the existence of external perturbation impacts the RS performance when users have different levels of preference biases, and the long-lasting impacts on RS after the removal of the external perturbations. We focus on one representative perturbation setting which is then expanded in the later experiments. Specifically, RS is perturbed during the first five time periods, during which 1% of all items have their predicted ratings artificially inflated by one additional star, capped at a maximum of five stars – this represents a short-term and small-scale perturbation in practice. We then keep simulating the user-system feedback loop for 100 time periods. Users have homogeneous preference bias level in the range of 0 to 1, with 6 equally-spaced values (i.e., 0, .2, .4, .6, .8, 1). We control for all other variables, including item features, users' preferences, consumption frequency, users' reliance on recommendations, etc.

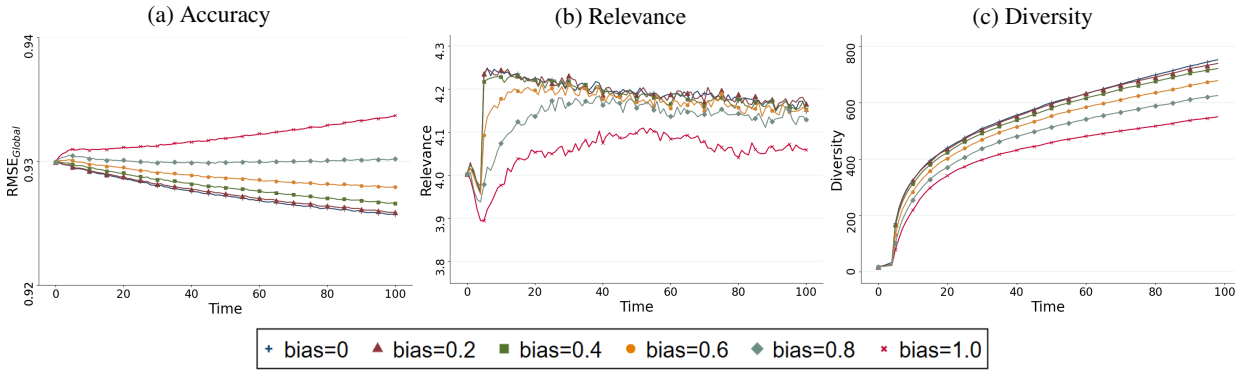
Experiment Results Figure 1 illustrates the longitudinal dynamics of the system's predictive performance and users' consumption outcomes under the interaction between the preference bias and external perturbations. The accuracy trends (Figure 1a) show that RMSE gradually improves under low preference bias regime ($bias < 0.8$) as data accumulates over time. However, RS performance often fails to improve with additional data when the preference bias is high ($bias \geq 0.8$) and even degrades when the preference bias is extreme ($bias = 1.0$). This indicates that the adverse effect of external perturbation stays strong when the preference bias is significant; and the extreme preference bias amplifies the negative consequence of external perturbation.

The relevance trends (Figure 1b) show that, during the first 5 periods, the consumption relevance is low and decreasing. This occurs because the perturbed items are artificially promoted by RS and ranked high in the recommendation lists, which increases their consumption probability during the perturbation period. Because these items are randomly selected and therefore only of average relevance to users, concentrated consumption of these items leads to low consumption relevance. Moreover, during the perturbation period, the interaction between the perturbation and high bias causes the relevance to decrease. Specifically, in the first timestep of the perturbation period, the perturbed items are consumed only due to the perturbation itself, while in subsequent periods they are consumed even more because of (i) the continuing perturbation *and* (ii)

Table 2: Summary of Experimental Application Settings

Dataset	Description	Users	Items	Ratings	Density
Sample of Netflix	Movie ratings from Netflix	3,000	3,000	344,021	3.82%
Sample of Yelp	Business ratings from Yelp	4,000	2,000	62,476	0.78%

Figure 1: Impact of Preference Bias and External Perturbation on Longitudinal RS Performance (Netflix, SVD)



the recommendation algorithms taking the distorted consumption data from the previous periods into account.

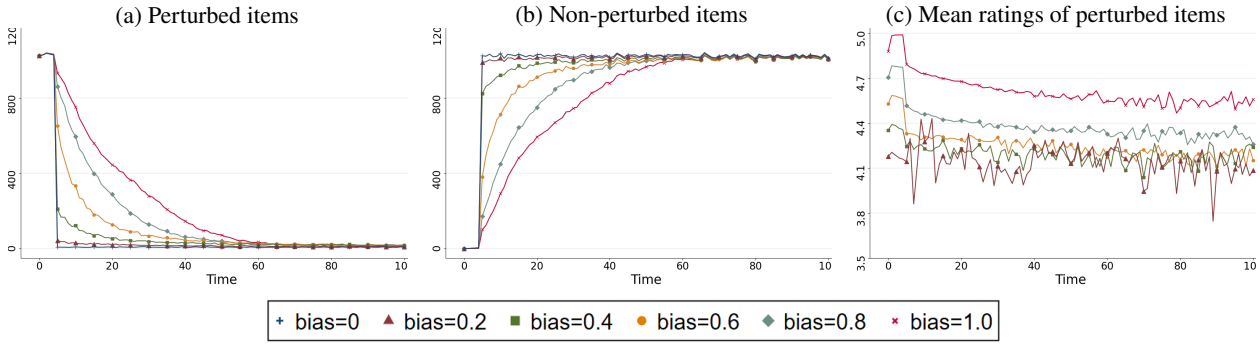
After the perturbation period, relevance gradually returns and converges to higher levels. The larger the preference bias, the longer the recovery takes. If users have low bias ($bias \leq 0.4$), the relevance recovers to a high level immediately after the perturbation period; but when users have high bias ($bias \geq 0.8$), the recovery of relevance can take much longer. As shown in Figure 1b, when preference bias is at 0.6, it takes SVD fewer than 30 timesteps after perturbation to recover to the expected, zero-bias relevance levels (around 4.2 stars), while it takes more than 35 timesteps to recover to a lower steady-state relevance level (around 4.1 stars) with preference bias at 1.0.

The diversity trends (Figure 1c) show low consumption diversity during the perturbation periods due to consumption concentrating on the perturbed items. After the perturbation, diversity gradually recovers but with different speeds and magnitudes under different levels of preference biases. Notably, when preference bias is small, the recovery in consumption diversity is swift (as evidenced by the steep slope of the rising diversity curve immediately after the perturbation period). The magnitude of recovery is also significantly larger under weaker preference biases; e.g., as shown in Figure 1c, consumption diversity with SVD reaches around 750 unique items by the end of the simulation period with no preference bias, while it remains below 550 with extreme preference bias. We observe highly consistent patterns on both RS performance and consumption outcomes with Yelp dataset and different RS algorithms as well (shown in Figures 7 and 8 of Appendix C).

To understand the underlying mechanism of the observed recommendation and consumption patterns, we further analyze users' consumption of the 1% perturbed items and the 99% non-perturbed items (see the first two panels in Figure 2). Without perturbation, the targeted set of items would have only a small chance of being consumed by users. However, during the perturbation window, these items are "promoted" and thus frequently recommended to users and get consumed. When the preference bias is high, the promotion of these items is reinforced by users through their biased post-consumption ratings submitted to RS. Those biased feedback ratings further contaminate the RS predictions for perturbed items and make them more likely to be recommended in the future, even long after the perturbation period ends, as shown in Figure 2a. The larger the bias degree, the longer it takes for the system to correct the abnormal consumption of the perturbed items. Meanwhile, Figure 2b shows that consumption of non-perturbed items is substantially suppressed during the perturbation period, and only recovers after the perturbation ends. Additionally, as shown in Figure 2c,

during perturbation, the average submitted ratings of the perturbed items are higher when bias is larger. Even long after the perturbation ends, the submitted ratings for the perturbed items remain much higher when $bias \geq 0.8$. These observations are highly consistent across different RS algorithms as shown in Figure 9 of Appendix D.

Figure 2: Consumption Analysis of Perturbed and Non-Perturbed Items (Netflix, SVD)



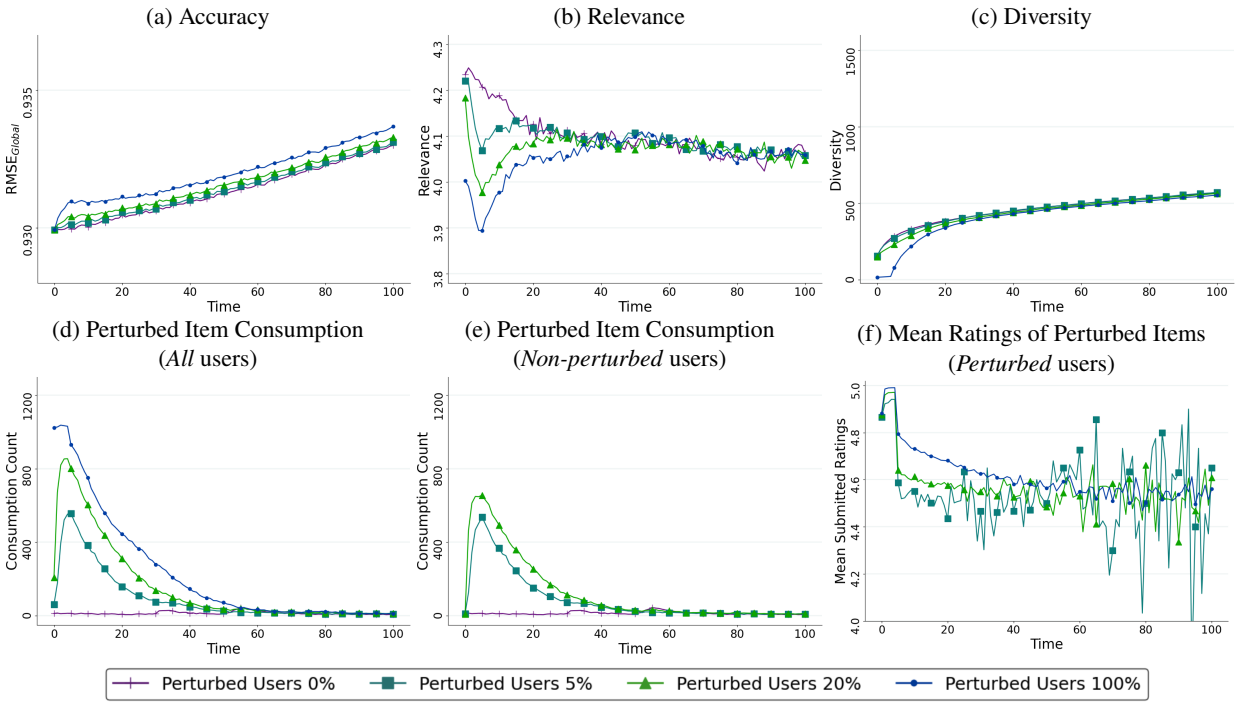
5 Experiment 2: Varying Perturbed User and Item Ratios

Experiment Setting Experiment 1 represents a small-scale perturbation setting, while in reality the ratio of perturbed users and items can vary. Therefore, in Experiment 2, we vary the proportion of perturbed users and items separately to understand how these variations impact the RS performance. First, we vary the portion of perturbed users as 0%, 5%, 20%, and 100% with preference bias 1. Second, we vary the portion of perturbed items as 0.1% and 1% with preference bias 1. All other parameters remain the same as in Experiment 1.

Varying Perturbed User Ratio Figure 3 shows how the impact of external perturbation varies with different portions of perturbed users. Specifically, Figures 3a-c show the overall system performance and consumption outcomes. Compared to when there is no perturbation (i.e., 0% perturbed users), even when only a small sample (e.g., 5% or 20%) of users are perturbed, there is a substantial impact on the system's predictive performance and consumption outcomes. The influence is non-linear in the size of the perturbed user sample. Also, after the perturbation ends, it takes significant time for consumption relevance to go back to the level when there is no perturbation. In terms of consumption diversity, the change is not substantial if only a small portion of users are perturbed, as the non-perturbed users still consume a substantial number of different items.

Figures 3d-f show a more detailed analysis of how the perturbed items are consumed across different user segments over time. When there is no perturbation, the 1% targeted items are rarely consumed and cannot get much "rating inflation" (as shown by the "Perturbed Users 0%" line in Figure 3d). Once a subset of users is perturbed, even as few as 5%, the average rating and consumption count of targeted items increase dramatically. When 20% of users are perturbed, the consumption count of targeted items across all users approaches that observed when 100% of users are perturbed (Figure 3d). The high consumption of targeted items persists for a substantial amount of time after the perturbation ends. Additionally, Figure 3e shows a substantial spillover effect of the external perturbation on non-perturbed users' consumption. Although these non-perturbed users do not receive the intentional item promotion, a large portion of them still end up

Figure 3: Robustness: Varying Perturbed User Ratio (Netflix, SVD)



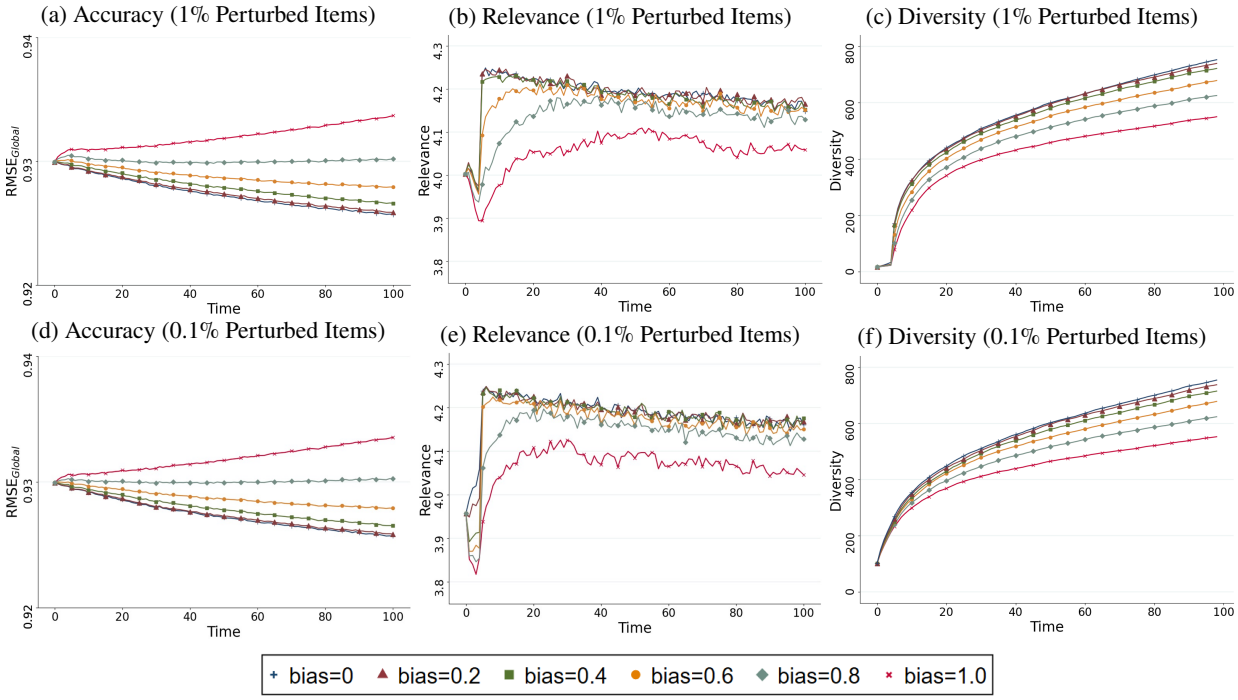
consuming the targeted items, because their recommendations and consequent consumption are affected by the perturbations on other users. This occurs because the inflated ratings submitted by perturbed users on targeted items substantially bias the system's predictions of these items for non-perturbed users and, thus, indirectly promote these targeted items to non-perturbed users. Finally, Figure 3f shows that the average submitted ratings for targeted items from perturbed users are highest during the perturbation period; afterwards, the average decreases to a lower level as RS recovers from the adversarial effect of perturbations. Unsurprisingly, RS recovers faster with fewer perturbed users.[5]

Varying Perturbed Item Ratio When varying the ratio of perturbed items as 1% vs. 0.1%, we observe no substantial differences in RMSE between the two perturbation conditions (Figures 4a and 4d). However, with fewer perturbed items (Figure 4e), consumption relevance drops well below 3.9 during the perturbation period, when preference bias is high (i.e., $bias \geq 0.6$). In contrast, with more perturbed items (Figure 4b), i.e., less concentrated perturbation, consumption relevance is almost always above 3.9. Intuitively, RS can learn the relative relevance of the perturbed items and recommend more relevant perturbed items to users. Thus, there is a larger chance for RS to recommend relevant items to users simply due to the inclusion of more items in the perturbed pool. Despite this milder negative effect within the perturbation period, less concentrated perturbation harms the system more severely in the post-perturbation period. Notably, consumption relevance recovers slower from a less concentrated perturbation at every bias level (Figures 4b and 4e), because having more perturbed items leads to more rating inflation, which takes longer for the system to correct in the

[5]The average submitted rating of perturbed users for targeted items exhibits larger variation (i.e., is noisier) when there are fewer consumptions of such items – i.e., when (a) there are fewer perturbed users and/or (b) towards the end of the simulation, after RS recovers from adversarial effects.

post-perturbation period. In terms of consumption diversity (Figures 4c and 4f), the impact is similar, except that, during the perturbation period, diversity is higher when the perturbed item ratio is lower. The reason is that more concentrated perturbation cannot fill all the top spots in the recommendation list with only a few perturbed items. Users still have opportunities to consume non-perturbed items, which adds to the consumption diversity.

Figure 4: Targeted Perturbation: 1% vs. 0.1% Perturbed Item Ratio (Netflix, SVD)

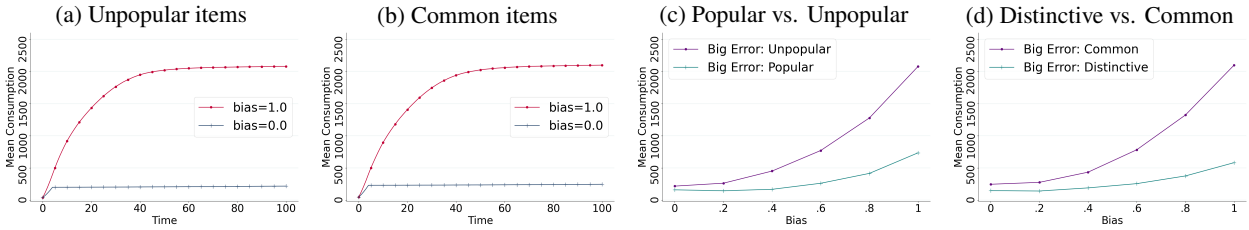


6 Experiment 3: Item Susceptibility to Perturbation and Preference Bias

Experiment Setting As discussed in Section 4, external perturbation inflates the system ratings of targeted items, and preference bias amplifies the inflation effect further. These two effects reinforce each other in the RS-user feedback loop, creating a snowball effect that adversely affects RS performance and consumption outcomes. However, it is not clear how and where this snowball effect accumulates. For example, are all items affected equally, or are some items more susceptible to the impacts of perturbation and preference bias? To answer this question, we categorize items into different groups and analyze the consumption patterns of each group. Since item consumption varies across different experiment settings, for fair comparison, we focus on items consumed during the first simulation period ($t = 1$), which remains identical across different bias conditions. We then track the consumption counts of these items over 100 periods under two bias conditions: 0.0 and 1.0. The external perturbation follows the same protocol as in Experiment 1, where we randomly select 1% of items and increase their displayed rating by one star during the first 5 periods.

To categorize all items consumed at $t = 1$, we first choose a group of "big-error" items based on the magnitude of *system error* (i.e., the difference between the system-displayed rating and the actual, "ground truth" user preference), since preference bias can only influence users' feedback when there is substantial

Figure 5: Mean Consumption of Big-Error Item Subgroups with Perturbations on All Users (Netflix, SVD)



system error. Then we further categorize the big-error items based on two common item features that have been shown to be relevant for preference bias propagation in RS (Zhou et al. 2024). The first categorization is by *popularity*, measured by the number of pre-existing item ratings at initialization. Items with a popularity score below the system-wide median are classified as “unpopular", while those exceeding the median are labeled as “popular". The second categorization is by *content distinctiveness*, defined by the number of other items that share similar content features. Specifically, we compute the cosine similarity of latent feature vectors (i.e., $ItemContent$) for each item pair. We then determine the number of neighbors an item has, where a neighbor is any item with a similarity score above 0.8. Using the median number of neighbors across all items in the system as a threshold, items with a high number of neighbors are classified as “common", while those with fewer neighbors are considered “distinctive".

Experiment Results Figure 5 provides an overview of the key analysis results. In particular, we find that the consumptions of unpopular and common items is substantially increased in the presence of external perturbations when bias is 1 as compared to when bias is 0 (Figures 5a and 5b). Moreover, we vary the magnitude of preference bias and compare the mean accumulated consumption counts during the entire simulation for different item subgroups (Figures 5c and 5d). When preference bias is more extreme, item heterogeneity manifests more strongly in terms of accumulated consumption count. When preference bias is absent, the difference between unpopular and popular items, or between common and distinctive items, is negligible in terms of accumulated consumption count. Overall, Figure 5 clearly shows that consumption of unpopular and common items is substantially inflated under external perturbations when bias is high. These observations are consistent across different RS algorithms, as shown in Figure 10 of Appendix E.

7 Robustness of the Main Findings

To establish the generality of our conclusions, we conduct numerous robustness checks regarding *external perturbations*, *user consumption and feedback*, and *simulation scope*. Our conclusions hold across all checks, and we discuss a few quantitative differences in the corresponding appendices.

External Perturbations. Holding all other parameters at their main-experiment values, we (i) vary the length of the perturbation window (Appendix F); (ii) vary the per-rating perturbation magnitude δ over the 1–5 rating scale (Appendix G); (iii) replace the random selection of perturbed items and perturbed users with alternative targeting rules and substitute the fixed rating perturbation with a minimal-promotion magnitude rule (Appendix H); and (iv) replace the abstract additive perturbation with two concrete, widely studied profile-injection attacks, namely the *average* and *bandwagon* attacks (Appendix I).

User Consumption and Feedback. Relaxing standard assumptions about the user population, we (i) vary the biased user ratio from 20% to 100% (Appendix J); (ii) replace the homogeneous preference-bias coefficient with a heterogeneous one drawn from a Beta distribution whose mean equals the target bias level (Appendix K); (iii) vary the recommendation reliance parameter α that governs how concentrated each user's consumption is on the top-ranked recommendations (Appendix L); and (iv) remove the Gaussian noise term from users' ground-truth preference ratings (Appendix M).

Simulation Scope and Statistical Significance. Varying setup choices that are peripheral to the modeling but could nevertheless affect quantitative outcomes, we (i) extend the simulation horizon from $T = 100$ to $T = 200$ time steps (Appendix N); (ii) scale the seeding dataset across 3,000, 10,000, and 30,000 users (Appendix O); and (iii) reproduce all main analyses across four recommendation algorithms including SVD, ItemKNN, LinUCB (Li et al. 2010), and MHCL (Li et al. 2025), and two datasets, Netflix and Yelp (Appendix C). We further verify that the cross-bias post-perturbation differences observed in the main experiment are statistically significant via timestep-level paired t-tests (Appendix P) and that they remain distinct relative to the run-to-run variability across the ten replications (Appendix Q).

Analytical Modeling Insights. Finally, in Appendix A, we validate and supplement our main findings by developing a tractable analytical model that derives closed-form expressions for the temporal dynamics of the attack influence (i.e., system contamination state) both during and after the attack window, providing a theoretical counterpart to the simulation-based evidence. Overall, the consistency of the findings strongly supports interpreting the bias-driven amplification of an external perturbation's adverse effects as a structural property of the user-RS feedback loop.

8 Strategic Attack Optimization under Preference Bias

To further highlight the strength of the interplay between external perturbations and preference biases, in this section we use a *prescriptive* framework to demonstrate how a strategic attacker, constrained by a finite total resource, can choose the different attack dimensions to maximize adverse impact to the platform. The analysis is based on the operational realities that an attacker faces. Specifically, resources, such as the number of fake accounts that can be activated, the attack window before detection, or the per-user manipulation cost are typically finite, so the attacker must allocate effort across different attack parameters. The framework we introduce below performs an optimization which allows examining, both quantitatively and structurally, how preference bias reshapes the attacker's optimal strategy and outcomes.

Attack Optimization Simulation. In this analysis, we model the attacker as choosing values for a set of three attack parameters $\{\rho, \delta, \tau\}$—user coverage ρ, perturbation magnitude δ, and attack duration τ—to optimize their attack outcome. Target items are fixed at a pre-defined percentage of the entire item population because the attacker often has a specific goal of promoting a selected set of items beforehand.[6] The attack outcome is defined as the cumulative perturbed-item consumption count $\Phi(\mathcal{U}_\pi^\rho, \delta, \tau)$ across all users over the simulation horizon, where $\mathcal{U}_\pi^\rho$ is the perturbed-user set that can be strategically chosen based on user coverage ρ.[7] We

[6] In the following experiment setups, we randomly select 1% of the item population as a fixed set of target (i.e., perturbed) items.
[7] For example, the attacker can prioritize to target users with a higher preference bias level to optimize their attack outcomes.

Algorithm 1: Optimization of attack parameters under an attack-effort constraint

Input :A fixed attack-effort level $E > 0$; the number of TPE trials N; a user targeting strategy (*random* or *bias-greedy*).

Initialize a TPE sampler; set the incumbent best attack outcome $\Phi^\star \leftarrow -\infty$;
for $n \leftarrow 1$ **to** N **do**
  Query TPE for a trial $\boldsymbol{\xi}^n$ subject to the effort constraint E;
  if *user targeting is random* **then**
    $\mathcal{U}_\pi^{\rho_n} \leftarrow$ a ρ^n-fraction of users drawn uniformly at random;
  else
    $\mathcal{U}_\pi^{\rho^n} \leftarrow$ the ρ^n-fraction of users with the highest preference bias;
  Evaluate $\Phi^n \leftarrow \Phi(\mathcal{U}_\pi^{\rho^n}, \delta^n, \tau^n)$ by simulation;
  Update the TPE sampler with the new trial–outcome pair $(\boldsymbol{\xi}^n, \Phi^n)$;
  if $\Phi^n > \Phi^\star$ **then** $\boldsymbol{\xi}^\star \leftarrow \boldsymbol{\xi}^n$; $\Phi^\star \leftarrow \Phi^n$;
return the best trial $\boldsymbol{\xi}^\star$ and its outcome $\Phi^\star$;

impose the effort constraint on the attack parameters multiplicatively, $E = \rho \times \delta \times \tau$, for a fixed effort level. This is motivated by standard a Cobb-Douglas-style trade-off (Cobb Douglas 1928) in which different attack parameters are modeled as complementary resources that constrain the attack activity.

Each evaluation of Φ requires a full simulation of the user-RS feedback loop, an expensive black-box query. We therefore optimize Φ with the Tree-structured Parzen Estimator (TPE; Bergstra et al. 2011, Akiba et al. 2019), a sequential Bayesian optimizer efficient in black-box optimization. It models the distribution of attack parameters conditioned on whether the attack outcome is "good" or "bad" (e.g., we consider the top 15% of outcomes "good" in the following experiment). Specifically, it first draws an initial set of attack parameters via random sampling from the prior distribution (e.g., a uniform distribution), to obtain a sufficient number of observed outcomes for density estimation. After this initialization phase, it splits all observed trials into "good" and "bad" groups based on the outcome threshold, and fits a kernel density estimate to each group. At each subsequent iteration, new attack parameters are proposed by drawing candidates from the "good" group and selecting the one that maximizes "good"-to-"bad" kernel density ratio. This process repeats iteratively, and the optimizer concentrates new proposals of attack parameters into the parameter space where the "good"-to-"bad" ratio is highest. We refer to each iteration of the parameter proposal as *trial* $\boldsymbol{\xi}^n = \{\rho^n, \delta^n, \tau^n\}$, defined as the realization of the attack parameters ρ, δ, and τ drawn by TPE at iteration n under the effort constraint. This optimization framework is intentionally general: the attacker may choose any subset of attack parameters to optimize and hold the remaining parameters fixed, with each fixed parameter assigned a predetermined value. As summarized in Algorithm 1, in each iteration $n \in \{1, \ldots, N\}$, TPE proposes trial $\boldsymbol{\xi}^n$ under the effort constraint, the corresponding perturbed-user set $\mathcal{U}_\pi^{\rho^n}$ is formed under either random or bias-greedy targeting, the attack outcome Φ^n is evaluated by simulation, and the sampler is refined with each trial-outcome pair. The procedure returns the best-performing trial $\boldsymbol{\xi}^\star$ and its attack outcome $\Phi^\star$.

Instantiation Setups. We instantiate the framework in two complementary setups. In both cases, we employ SVD on Netflix dataset as the recommender, consider a time horizon of $T = 100$ periods with the attack starting at $t = 0$. The first instantiation is the *two-free-parameter setup*, which fixes the perturbation magnitude

Table 3: Optimal attack outcome (cumulative perturbed-item consumption) under an effort budget of $E = 5$

		Optimal attack outcome	
User preference bias	User targeting	Two-free-parameter	Three-free-parameter
Heterogeneous	Bias-greedy	12,778 (+72.7%)	13,204 (+46.9%)
	Random	10,668 (+44.2%)	11,644 (+29.6%)
Zero (baseline)	Random	7,397	8,988

at $\delta = 1$ (one rating star) and treats only user coverage ρ and attack duration τ as optimizable parameters under effort constraint $E = \rho \cdot \tau = 5$, (e.g., if the attacker chooses to attack the entire population $\rho = 1$, the attack can sustain for $\tau = 5$ days at maximum), with $N = 50$ TPE trials. By holding the perturbation magnitude constant, this setup isolates the coverage-versus-duration trade-off and serves as a reference point. The second, more comprehensive instantiation is the *three-free-parameter setup*, which makes all three attack parameters optimizable and uses the same effort constraint $E = \rho \cdot \delta \cdot \tau = 5$, (e.g., if the attacker chooses to attack the entire user population with 0.5-star perturbation magnitude, the attack can sustain 10 days at maximum), with $N = 100$ TPE trials.

Experiment Conditions. Within each setup we run three contrasting conditions: (i) *random targeting*, where user-level preference bias is drawn heterogeneously from a Beta distribution with mean 0.35 and standard deviation 0.1 and the perturbed user set is sampled uniformly at random;[8] (ii) *bias-greedy targeting*, where user-level preference bias is the same as in the previous condition, but the perturbed user set consists of the users with the highest bias; and (iii) a *random-targeting zero-bias baseline* scenario where every user has zero preference bias and targeting is random.

Findings. Table 3 reports the optimal attack outcome for the combination of two instantiation setups and three targeting strategies described above. There are two key observations. First, the amplification of the attack effect is enhanced as more parameters are allowed in the optimization. Notably, going from two- to three-free-parameter optimization setup (under the same attack effort constraint) further increases consumption of perturbed items significantly across all conditions. Second, and more importantly, preference bias makes the platform structurally more vulnerable to an effort-optimizing attacker: the optimal attack outcome is consistently and substantially higher when the user population exhibits preference bias as compared to the zero-bias baseline. Moreover, a bias-aware attacker can amplify this advantage substantially: once the attacker actively targets the highest-bias users, the attack outcome improves by 72.7% and 46.9% (relative to the zero-bias baseline) under two- and three-free-parameter setups, respectively. In other words, preference bias not only improves the attack effectiveness, but it also generates a vulnerable structure that a strategic adversary can exploit to inflict extra harm. As an additional illustration, Figure 6 displays the trial clouds created by the Bayesian optimization processes in TPE with the two-free-parameter setup for the three conditions, which shows that the differences in attack outcomes for these conditions are highly and systematically substantial across the entire parameter space.

[8] This is simulated to reflect a right-skewed preference bias distribution empirically observed in prior literature (Adomavicius et al. 2013).

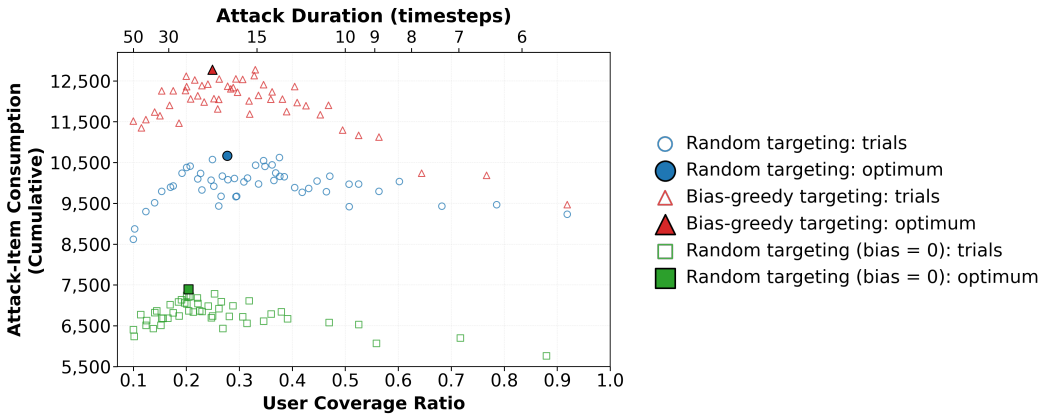


Figure 6: TPE trials of two-free-parameter optimization

Taken together, these results reinforce that preference bias is a structural vulnerability of the user-RS feedback loop – preference bias amplifies the impact caused by external perturbations, and an adversary can strategically exploit this phenomenon further using limited attack resources.

9 Discussion and Conclusions

This research demonstrates that user preference bias amplifies and prolongs the adverse effects of external perturbations on RS. Through simulations across multiple recommendation algorithms and datasets, we show that perturbation effects intensify with stronger bias, transforming typically short-lived shocks into persistent phenomena that degrade predictive accuracy and consumption outcomes even after the perturbation ends. Perturbing only a portion of users can induce system-wide effects nearly as severe as perturbing the entire population, because biased ratings from perturbed users distort predictions for others, causing widespread spillover. At the item level, unpopular items and content-wise common items are disproportionately vulnerable to rating inflation under preference bias. Furthermore, we perform comprehensive robustness checks on key aspects of our simulation modeling based on different characteristics of external perturbations, user population, recommendation process, and simulation scope; the main findings are consistent across extensive variations and are also supported and complemented by an analytical model of bias-driven contamination dynamics. We also examine how a strategic attacker can exploit these dynamics by jointly optimizing user coverage, attack duration, and perturbation magnitude under a fixed effort constraint. We demonstrate that preference bias not only significantly enhances the attack effectiveness, but also becomes a lever that can be exploited to inflict additional adverse impact.

These findings carry direct implications across the many contexts in which RS are deployed, where the consequences of leaving the bias-driven amplification mechanism unaddressed differ in kind and severity by domain. In high-revenue commercial contexts, such as e-commerce, online advertising, and app-store ranking, the persistence of contamination after adversarial attack turns even small-scale adversarial activity into a high-ROI manipulation channel; here platform auditing and red-teaming protocols should evaluate not only the immediate injection effect but also the long-run residual distortion in honest users' submitted ratings. In

content- and cultural-consumption contexts, such as streaming, music, and news platforms, the bias-amplified suppression of consumption diversity that we document means that even short-lived perturbations can systematically marginalize niche content, emerging creators, and underexposed items, with corresponding fairness and welfare consequences. In high-stakes domains, such as healthcare, financial-product, and educational recommendation, individual recommendations carry safety, monetary, or equity consequences. In these settings, the post-attack persistence of contamination raises governance and liability concerns and motivates the adoption of conservative deployment practices and audit trails, especially because the residual distortion can outlast detection windows and is produced by honest users' biased feedback rather than by easily flagged fake user profiles. Cutting across these domains, our findings also suggest that robust-RS evaluation and benchmarking practices should test the *longitudinal* resilience of recommender systems rather than focus exclusively on one-shot prediction accuracy, and they motivate the design of post-attack recovery protocols that explicitly encompass mitigating the residual contamination of honest users' submitted ratings rather than merely removing the fake user profiles.

This work highlights the need to redesign perturbation-mitigation methods in a way that explicitly accounts for user preference bias, suggesting *hybrid* strategies to attack defense rather than any single remedy. As a starting point, we present an overview of possible mitigation measures that have been explored in prior work and can be taken as components of potential hybrid approaches. In particular, Table 4 summarizes the existing mitigation approaches along two complementary axes. The first axis is "mitigation target": *attack mitigation* measures harden the system against external perturbations, whereas *bias mitigation* measures are designed to reduce/prevent preference bias. The second axis is "mitigation timing": *proactive* measures are designed towards preventing the adverse effects from occurring to begin with, whereas *reactive* measures detect and repair the attack effects after the fact. Crossing the two axes yields a four-quadrant taxonomy, an overview of which is provided in Table 4, with concrete instantiations and supporting literature. It is important to reiterate that the aforementioned measures have been designed to focus on a specific purpose: attack vs. bias mitigation. However, the unique interplay between external perturbations and preference biases, i.e., the fact that preference bias can propagate a transient perturbation into persistent, amplified contamination, highlights the need to investigate various hybrid approaches that pair attack mitigation with bias mitigation, possibly across both the proactive and reactive stages.

More broadly, two follow-up directions naturally extend this work. First, thoughtful adaptation and/or combination of different mitigation approaches (discussed in Table 4) represent promising directions for future study in the unique context of the perturbation-bias interplay. Second, more generally, this study calls for *hybrid* and *longitudinal* mitigation approaches to external attacks on recommender systems. This approach should not only understand and consider how the external attacks affect the users' consumption at a given time, but also recognize that the entire consumption trajectories can be significantly affected after the attack, especially in the presence of behavioral biases (such as preference bias). Together, these directions point towards a comprehensive and general framework of bias-aware attack defense in practice.

Table 4: An overview of mitigation measures, by target (attack vs. bias) and timing (proactive vs. reactive).

	Attack mitigation	**Bias mitigation**
Proactive mitigation	*Sybil-/shilling-attack prevention* that blocks fake-account creation and profile injection before contaminated ratings ever enter the system, using information security measures such as account- or identity-verification (Lam Riedl 2004, Feher et al. 2012, Chirra 2022, Yusop et al. 2025).	*Rating-interface redesign* that attenuates preference bias at its source, for example, de-emphasizing displayed rating predictions by using a graphical recommendation display (Adomavicius et al. 2019) and presenting a top-N list without predicted numerical rating (Adomavicius et al. 2022b).
Reactive mitigation	*Machine unlearning* that removes the influence of attacker-injected, contaminated records from the trained model, paired with *attack detection* extended to flag the manipulation footprint after the attack (Cao Yang 2015, Ginart et al. 2019, Bourtoule et al. 2021, Chen et al. 2022). *Adversarially robust RS algorithms* (e.g., robust matrix factorization, adversarial training) that bound the leverage of any injected ratings that evade fake-account/profile blocks (Chen et al. 2024, Nguyen et al. 2024, Zhang et al. 2025).	*Post-hoc debiasing* at the data or algorithm level, for example, by estimating the user preference bias and debiasing the submitted rating (Zhou et al. 2024) and by designing bias-aware recommender systems (Zhang et al. 2019). *Rating re-elicitation* that prompts users to re-rate items without displayed predictions, replacing contaminated ratings with fresh, unbiased ratings (Cosley et al. 2003, Amatriain et al. 2009b, Krishnan et al. 2014).

References

Abdollahpouri H (2019) Popularity bias in ranking and recommendation. *Proc. of the 2019 AAAI/ACM Conf. on AI, Ethics, and Society*, 529–530.

Abdollahpouri H, Mansoury M, Burke R, Mobasher B (2019) The unfairness of popularity bias in recommendation. *arXiv preprint arXiv:1907.13286* .

Abdollahpouri H, Mansoury M, Burke R, Mobasher B (2020) The connection between popularity bias, calibration, and fairness in recommendation. *Proc. of the 14th ACM conf. on recommender systems*, 726–731.

Abdollahpouri H, Mansoury M, Burke R, Mobasher B, Malthouse E (2021) User-centered evaluation of popularity bias in recommender systems. *29th ACM conf. on user modeling, adaptation and personalization*, 119–129.

Adomavicius G, Bockstedt J, Curley S, Zhang J (2022a) Recommender systems, ground truth, and preference pollution. *AI Magazine* 43(2):177–189.

Adomavicius G, Bockstedt JC, Curley SP, Zhang J (2013) Do recommender systems manipulate consumer preferences? a study of anchoring effects. *Information Systems Research* 24(4):956–975.

Adomavicius G, Bockstedt JC, Curley SP, Zhang J (2018) Effects of online recommendations on consumers' willingness to pay. *Information Systems Research* 29(1):84–102.

Adomavicius G, Bockstedt JC, Curley SP, Zhang J (2019) Reducing recommender system biases: an investigation of rating display designs1. *MIS Quarterly* 43(4):1321–1341.

Adomavicius G, Bockstedt JC, Curley SP, Zhang J (2022b) Effects of personalized recommendations versus aggregate ratings on post-consumption preference responses. *Mis Quarterly* 46(1):627–644.

Adomavicius G, Kwon Y (2014) Optimization-based approaches for maximizing aggregate recommendation diversity. *INFORMS Journal on Computing* 26(2):351–369.

Akiba T, Sano S, Yanase T, Ohta T, Koyama M (2019) Optuna: A next-generation hyperparameter optimization framework. *Proc. of the 25th ACM SIGKDD Intl. Conf. on Knowledge Discovery & Data Mining (KDD '19)*.

Amatriain X, Pujol JM, Oliver N (2009a) I like it... i like it not: Evaluating user ratings noise in recommender systems. *Intl. Conf. on User Modeling, Adaptation, and Personalization*, 247–258 (Springer).

Amatriain X, Pujol JM, Tintarev N, Oliver N (2009b) Rate it again: increasing recommendation accuracy by user re-rating. *Proc. of the Third ACM Conf. on Recommender Systems*, 173–180 (Association for Computing Machinery).

Bennett J, Lanning S (2007) The netflix prize. *Proc. of KDD cup and workshop*, volume 2007, 35.

Bergstra J, Bardenet R, Bengio Y, Kégl B (2011) Algorithms for hyper-parameter optimization. *Advances in Neural Information Processing Systems 24 (NeurIPS 2011)*, 2546–2554 (Curran Associates, Inc.).

Bourtoule L, Chandrasekaran V, Choquette-Choo CA, Jia H, Travers A, Zhang B, Lie D, Papernot N (2021) Machine unlearning. *2021 IEEE Symposium on Security and Privacy (SP)*, 141–159 (IEEE).

Burke R, Mobasher B, Williams C, Bhaumik R (2006) Classification features for attack detection in collaborative recommender systems. *ACM SIGKDD Intl. Conf. on Knowledge discovery and data mining*, 542–547.

Cao Y, Chen X, Yao L, Wang X, Zhang WE (2020) Adversarial attacks and detection on reinforcement learning-based interactive recommender systems. *ACM SIGIR conf. on research and development in information retrieval*.

Cao Y, Yang J (2015) Towards making systems forget with machine unlearning. *2015 IEEE Symposium on Security and Privacy*, 463–480 (IEEE).

Carare O (2012) The impact of bestseller rank on demand: Evidence from the app market. *Intl. Economic Review* .

Chaney AJ, Stewart BM, Engelhardt BE (2018) How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. *Proc. of the 12th ACM conf. on recommender systems*, 224–232.

Chapman GB, Johnson EJ (2002) Incorporating the irrelevant: Anchors in judgments of belief and value. Gilovich T, Griffin D, Kahneman D, eds., *Heuristics and Biases: The Psychology of Intuitive Judgment*, 120–138.

Chen C, Sun F, Zhang M, Ding B (2022) Recommendation unlearning. *Proc. of the ACM Web Conf. 2022*, 2768–2777.

Chen J, Dong H, Wang X, Feng F, Wang M, He X (2023) Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems* 41(3):1–39.

Chen J, Zhang L, Yang N (2024) Improving adversarial robustness for recommendation model via cross-domain distributional adversarial training. *Proc. of the 18th ACM Conf. on Recommender Systems*, 278–286.

Chirra BR (2022) Strengthening cybersecurity with behavioral biometrics: Advanced authentication techniques. *Intl. Journal of Advanced Engineering Technologies and Innovations* 1(3):273–294.

Cobb CW, Douglas PH (1928) A theory of production. *The American economic review* 18(1):139–165.

Collins A, Tkaczyk D, Aizawa A, Beel J (2018) Position bias in recommender systems for digital libraries. *Intl. Conf. on Information*, 335–344 (Springer).

Cosley D, Lam SK, Albert I, Konstan JA, Riedl J (2003) Is seeing believing? how recommender system interfaces affect users' opinions. *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, 585–592.

Deldjoo Y, Noia TD, Merra FA (2021) A survey on adversarial recommender systems: from attack/defense strategies to generative adversarial networks. *Acm Computing Surveys (Csur)* 54(2):1–38.

Di Noia T, Malitesta D, Merra FA (2020) Taamr: Targeted adversarial attack against multimedia recommender systems. *2020 50th Annual IEEE/IFIP Intl. Conf. on dependable systems and networks workshops (DSN-W)*, 1–8 (IEEE).

Epley N, Gilovich T (2010) Anchoring unbound. *Journal of Consumer Psychology* 20(1):20–24.

Fan W, Zhao X, Li Q, Derr T, Ma Y, Liu H, Wang J, Tang J (2023) Adversarial attacks for black-box recommender systems via copying transferable cross-domain user profiles. *IEEE Trans. on Knowledge and Data Engineering* .

Feher C, Elovici Y, Moskovitch R, Rokach L, Schclar A (2012) User identity verification via mouse dynamics. *Information Sciences* 201:19–36.

Funk S (2006) Netflix update: Try this at home. `https://sifter.org/~simon/journal/20061211.html`, accessed October 10, 2020.

Ge Y, Liu S, Fu Z, Tan J, Li Z, Xu S, Li Y, Xian Y, Zhang Y (2024) A survey on trustworthy recommender systems. *ACM Transactions on Recommender Systems* 3(2):1–68.

Ge Y, Zhao S, Zhou H, Pei C, Sun F, Ou W, Zhang Y (2020) Understanding echo chambers in e-commerce recommender systems. *ACM SIGIR conf. on research and development in information retrieval*, 2261–2270.

Ginart A, Guan M, Valiant G, Zou JY (2019) Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems* 32.

Gomez-Uribe CA, Hunt N (2015) The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)* 6(4):1–19.

Granka LA, Joachims T, Gay G (2004) Eye-tracking analysis of user behavior in WWW search. *Proc. of the 27th Annual Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 478–479.

Gunes I, Kaleli C, Bilge A, Polat H (2014) Shilling attacks against recommender systems: a comprehensive survey. *Artificial Intelligence Review* 42:767–799.

Hale J (2021) Tiktok's recommendation algorithm is even more powerful—and potentially dangerous—than youtube's. URL `https://www.tubefilter.com/2021/07/21/tiktok-recommendation-algorithm-wall-street-journal-guillaume-chaslot/`, accessed: 2025-02-22.

Hofmann K, Schuth A, Bellogin A, De Rijke M (2014) Effects of position bias on click-based recommender evaluation. *European Conf. on Information Retrieval*, 624–630 (Springer).

Joachims T, Granka L, Pan B, Hembrooke H, Gay G (2005) Accurately interpreting clickthrough data as implicit feedback. *Proc. of the 28th Annual Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR '05)*.

Joachims T, Granka L, Pan B, Hembrooke H, Radlinski F, Gay G (2007) Evaluating the accuracy of implicit feedback from clicks and query reformulations in Web search. *ACM Transactions on Information Systems (TOIS)* .

Klimashevskaia A, Jannach D, Elahi M, Trattner C (2024) A survey on popularity bias in recommender systems. *User Modeling and User-Adapted Interaction* 34(5):1777–1834.

Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. *Proc. of the 14th ACM SIGKDD intl. conf. on Knowledge discovery and data mining*, 426–434.

Krishnan S, Patel J, Franklin MJ, Goldberg K (2014) A methodology for learning, analyzing, and mitigating social influence bias in recommender systems. *Proc. of the 8th ACM Conf. on Recommender Systems*, 137–144.

Lam SK, Riedl J (2004) Shilling recommender systems for fun and profit. *Proc. of the 13th intl. conf. on World Wide Web*, 393–402.

Li H, Di S, Chen L (2022) Revisiting injective attacks on recommender systems. *Advances in Neural Information Processing Systems* 35:29989–30002.

Li L, Chu W, Langford J, Schapire RE (2010) A contextual-bandit approach to personalized news article recommendation. *Proc. of the 19th Intl. Conf. on World Wide Web (WWW '10)*, 661–670 (ACM).

Li X, Shui C, Zhao Z, Dong J, Yu Y (2025) Multi-channel hypergraph contrastive learning for matrix completion. *ACM Transactions on Information Systems* 44(1):1–27.

Liu D, Cheng P, Dong Z, He X, Pan W, Ming Z (2020) A general knowledge distillation framework for counterfactual recommendation via uniform data. *ACM SIGIR conf. on research and development in information retrieval*.

McKinsey (2013) How retailers can keep up with consumers. URL `https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers`, accessed: 2025-02-22.

Mobasher B, Burke R, Bhaumik R, Williams C (2007) Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology (TOIT)* 7(4):23–es.

Newaz AI, Haque NI, Sikder AK, Rahman MA, Uluagac AS (2020) Adversarial attacks to machine learning-based smart healthcare systems. *GLOBECOM 2020-2020 IEEE Global Communications Conf.*, 1–6 (IEEE).

Nguyen TT, Quoc Viet Hung N, Nguyen TT, Huynh TT, Nguyen TT, Weidlich M, Yin H (2024) Manipulating recommender systems: A survey of poisoning attacks and countermeasures. *ACM Computing Surveys* 57(1):1–39.

O'Mahony M, Hurley N, Kushmerick N, Silvestre G (2004) Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technology (TOIT)* 4(4):344–377.

Ovaisi Z, Ahsan R, Zhang Y, Vasilaky K, Zheleva E (2020) Correcting for selection bias in learning-to-rank systems. *Proc. of The Web Conf. 2020*, 1863–1873.

Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. *The 10th intl. Conf. on World Wide Web*, 285–295.

Shehmir S, Kashef R (2026) RoLLMRec: a robust LLM-based recommender system for defending against shilling and prompt injection attacks. *Frontiers in Computer Science* 8:1735253.

Si M, Li Q (2020) Shilling attacks against collaborative recommender systems: a review. *Artificial Intelligence Rev.* .

Steck H (2011) Item popularity and recommendation accuracy. *Proc. of the fifth ACM conf. on Recommender systems*.

Sukiennik N, Gao C, Li N (2024) Uncovering the deep filter bubble: narrow exposure in short-video recommendation. *Proc. of the ACM Web Conf. 2024*, 4727–4735.

Tang J, Wen H, Wang K (2020) Revisiting adversarially learned injection attacks against recommender systems. *Proc. of the 14th ACM Conf. on Recommender Systems*, 318–327.

Tversky A, Kahneman D (1974) Judgment under uncertainty: Heuristics and biases. *Science* 185(4157):1124–1131.

Ursu RM (2018) The power of rankings: Quantifying the effect of rankings on online consumer search and purchase decisions. *Marketing Science* 37(4):530–552.

Wang S, Zhang X, Wang Y, Ricci F (2024) Trustworthy recommender systems. *ACM Transactions on Intelligent Systems and Technology* 15(4):1–20.

Wang X, Zhang R, Sun Y, Qi J (2021) Combating selection biases in recommender systems with a few unbiased ratings. *Proc. of the 14th ACM Intl. Conf. on Web Search and Data Mining*, 427–435.

Wang Z, Zhang Z, Dong J, Wang K (2026) Causality-aware dual-scale preference model for sequential recommendation. *Expert Systems with Applications* 313:131484.

Yusop MIM, Kamarudin NH, Suhaimi NHS, Hasan MK (2025) Advancing passwordless authentication: A systematic review of methods, challenges, and future directions for secure user identity. *IEEE Access* 13:13919–13943.

Zhang J, Adomavicius G, Gupta A, Ketter W (2020) Consumption and performance: Understanding longitudinal dynamics of recommender systems via an agent-based simulation framework. *Information Systems Research* .

Zhang K, Cao Q, Sun F, Wu Y, Tao S, Shen H, Cheng X (2025) Robust recommender system: a survey and future directions. *ACM Computing Surveys* 58(1):1–38.

Zhang R, Meng J, Sun Y, Xu Z, Yin B, Li H, Zhang Y, Wang C (2026) MCLMR: A model-agnostic causal learning framework for multi-behavior recommendation. *Proc. of the ACM Web Conf. 2026*, 6481–6492 (ACM).

Zhang X, Chen J, Zhang R, Wang C, Liu L (2021) Attacking recommender systems with plausible profile. *IEEE Transactions on Information Forensics and Security* 16:4788–4800.

Zhang X, Xie H, Zhao J, Lui JCS (2019) Understanding assimilation-contrast effects in online rating systems: Modelling, debiasing, and applications .

Zhou M, Zhang J, Adomavicius G (2024) Longitudinal impact of preference biases on recommender systems' performance. *Information Systems Research* 35(4):1634–1656.

Zhu Z, He Y, Zhao X, Caverlee J (2021) Popularity bias in dynamic recommendation. *Proc. of the 27th ACM SIGKDD conf. on knowledge discovery & data mining*, 2439–2449.

Appendix A Analytical Model of Bias-Driven Contamination Dynamics

This appendix develops a tractable closed-form model that explains how the preference bias amplifies and perpetuates the impact of an external perturbation attack on a recommender system (RS). The model formalizes the self-reinforcing feedback loop highlighted in the main paper, and yields explicit expressions for the time path of system distortion during and after the attack.

A.1 Setup and Notation

Let $\mathcal{U}$ and $\mathcal{I}$ denote the user and item sets, with discrete time periods $t = 0, 1, 2, \ldots$. For each $(u, i) \in \mathcal{U} \times \mathcal{I}$, $r_{ui} \in [r_{\min}, r_{\max}]$ is user u's latent *true preference* for item i (unobserved by the RS), P^t_{ui} is the RS's *predicted preference* at the start of period t (computed based on all ratings submitted through period $t-1$), $\tilde{P}^t_{ui}$ is the prediction *displayed* to user u (which may differ from P^t_{ui} under attack), and S_{ui} is the rating user u submits upon consuming item i.

Preference bias. We use the same functional form of preference bias as stated in Equation (3) of the main paper, where submitted ratings linearly anchors toward the displayed prediction:

$$S_{ui} \;=\; r_{ui} + b\big(\tilde{P}^t_{ui} - r_{ui}\big) \;\Longleftrightarrow\; S_{ui} - r_{ui} \;=\; b\big(\tilde{P}^t_{ui} - r_{ui}\big), \tag{7}$$

where $b \in [0, 1]$ is the (homogeneous) preference-bias parameter.[9] With $b = 0$, users always report their true preferences; with $b = 1$, they perfectly mimic the displayed prediction; intermediate b produces partial anchoring.

Consumption. The probability that a user consumes the item displayed at rank i is $\Pr(\text{cons.} \mid \text{rank} = i) = (\alpha - 1)\alpha^{-i}$ for some rank-decay factor $\alpha > 1$, so the maximum consumption probability (at top-1 rank) is $p_{\max} = (\alpha - 1)/\alpha$. Items at higher ranks are exponentially less likely to be consumed and rated; promoting an item up the recommendation list therefore directly increases the rate at which it acquires ratings.

Attack. The adversary selects target items $\mathcal{I}_\pi \subseteq \mathcal{I}$, target users $\mathcal{U}_\pi \subseteq \mathcal{U}$, and an attack window $\mathcal{T}_\pi = \{1, \ldots, \tau\}$. For each $(u, i, t) \in \mathcal{U}_\pi \times \mathcal{I}_\pi \times \mathcal{T}_\pi$, the displayed prediction is inflated by a constant $\delta > 0$: $\tilde{P}^t_{ui} = P^t_{ui} + \delta$; otherwise $\tilde{P}^t_{ui} = P^t_{ui}$. We write $\rho = |\mathcal{U}_\pi|/|\mathcal{U}|$ for the fraction of perturbed users.

A.2 Contamination State and its Temporal Dynamics

To track the cumulative distortion of attacked items, define the *contamination state*

$$\theta_t \;=\; \frac{1}{|\mathcal{I}_\pi|\,|\mathcal{U}|} \sum_{i \in \mathcal{I}_\pi} \sum_{u \in \mathcal{U}} \big(P^t_{ui} - r_{ui}\big), \qquad \theta_0 = 0, \tag{8}$$

as the population-averaged amount by which the RS overpredicts true preferences for attacked items at period t.

Temporal Contamination Dynamics We use a recurrence relation to model the temporal dynamics of the contamination state. Specifically, the contamination state at timestep $t+1$ is a linear combination of the prior contamination state (at timestep t) and the average contamination amount introduced in the most recent iteration, i.e., there exists a system-level *learning rate* $\lambda \in (0, 1)$ such that

$$\theta_{t+1} \;=\; (1 - \lambda)\,\theta_t + \lambda\,\bar{c}_t. \tag{9}$$

[9] In the main paper, we denote this preference bias parameter as *bias*. For notational simplicity, we abbreviate it further as b in this appendix.

Here $\bar{c}_t$ is the *contamination influx*, i.e., the amount of contamination being introduced at the current timestep t. Specifically, $\bar{c}_t$ is the consumption-weighted mean excess submitted rating $S_{ui} - r_{ui}$ for attacked items in period t (please see A.3. for the full derivation of this quantity).

The learning rate λ reflects how fast any contamination influx is incorporated into the contamination state, and it is controlled by the learning pace of RS, i.e., how aggressively new ratings overwrite existing rating history during retraining of RS: a small λ corresponds to a slow, inertial RS, while a large λ corresponds to one that rapidly tracks recent feedback. When the learning pace is large, the RS tends to reflect new ratings (or contaminated new ratings) into its prediction (or the contamination state).

By the recurrence relation stated in Equation 9, we assume that the contamination state dynamically changes with the new contamination influx in a linear manner. Our theoretical arguments derived below can generalize to any RS algorithms that make this assumption hold. For example, one such simple and realistic RS algorithm is the item mean predictor that calculates the predicted rating as a running average of the submitted rating in an item-specific manner.

To understand the relationships between the contamination state θ_t and time t, preference bias b, and other important parameters, we want to derive the explicit expression for θ_t. To do that, we first derive the explicit expression for $\bar{c}_t$, as it is a key factor in the recurrence relation for θ_t stated in Equation (9).

A.3 Derivation of Contamination Influx $\bar{c}_t$

We derive an explicit expression for $\bar{c}_t$ (in terms of preference bias and other underlying parameters) in two regimes (i.e., within and after the attack window). Let p_0 denote the natural (baseline) consumption probability of an attacked item by a non-perturbed user (i.e., the probability at the rank the item would occupy in the absence of perturbation), and let $p(\delta, \alpha) \in (p_0, p_{\max}]$ denote the consumption probability of an attacked item by a perturbed user. By the rank-decay form of the consumption model in Section A.1, $p(\delta, \alpha)$ is strictly increasing in δ and saturates at $p_{\max} = (\alpha - 1)/\alpha$ once the inflated item reaches rank 1.

Active-attack regime ($t \in \mathcal{T}_\pi$). Contamination influx $\bar{c}_t$ is the average per-pair contribution to the period-t rating corpus, with each $(u, i) \in \mathcal{U}_\pi \times \mathcal{I}_\pi$ pair weighted by its consumption probability. Within the attack window, perturbed users see attacked items at the inflated rank and consume them with probability $p(\delta, \alpha)$, while unperturbed users see the natural rank and consume them with probability p_0. Splitting the population sum by perturbation status,

$$\bar{c}_t \;=\; \frac{1}{|\mathcal{I}_\pi|\,|\mathcal{U}|}\left[\, p(\delta,\alpha)\sum_{i\in\mathcal{I}_\pi}\sum_{u\in\mathcal{U}_\pi}(S_{ui}-r_{ui}) \;+\; p_0\sum_{i\in\mathcal{I}_\pi}\sum_{u\notin\mathcal{U}_\pi}(S_{ui}-r_{ui})\right]. \tag{10}$$

The bias-anchoring identity (7) factors b out of the sum, and the perturbation rule $\tilde{P}^t_{ui} = P^t_{ui} + \delta$ on the perturbed pairs (and $\tilde{P}^t_{ui} = P^t_{ui}$ elsewhere) isolates the δ shift:

$$\begin{aligned}
\bar{c}_t \;&=\; \frac{b}{|\mathcal{I}_\pi|\,|\mathcal{U}|}\left[\, p(\delta,\alpha)\sum_{i\in\mathcal{I}_\pi}\sum_{u\in\mathcal{U}_\pi}\big(P^t_{ui}+\delta-r_{ui}\big) \;+\; p_0\sum_{i\in\mathcal{I}_\pi}\sum_{u\notin\mathcal{U}_\pi}\big(P^t_{ui}-r_{ui}\big)\right] \\
&=\; \frac{b}{|\mathcal{I}_\pi|\,|\mathcal{U}|}\left[\, p(\delta,\alpha)\underbrace{\sum_{i\in\mathcal{I}_\pi}\sum_{u\in\mathcal{U}_\pi}\big(P^t_{ui}-r_{ui}\big)}_{=\,\rho\,|\mathcal{I}_\pi|\,|\mathcal{U}|\,\theta_t} \;+\; p(\delta,\alpha)\,|\mathcal{I}_\pi|\,|\mathcal{U}_\pi|\,\delta \;+\; p_0\underbrace{\sum_{i\in\mathcal{I}_\pi}\sum_{u\notin\mathcal{U}_\pi}\big(P^t_{ui}-r_{ui}\big)}_{=\,(1-\rho)\,|\mathcal{I}_\pi|\,|\mathcal{U}|\,\theta_t}\right] \\
&=\; b\big[\rho\,p(\delta,\alpha)+(1-\rho)\,p_0\big]\theta_t \;+\; b\,\rho\,p(\delta,\alpha)\,\delta.
\end{aligned} \tag{11}$$

The two underbraces invoke the *random user perturbation* assumption: $\mathcal{U}_\pi$ is a uniform random subset of $\mathcal{U}$, so the average prediction excess for attacked items is the same in each subgroup and equal to the population average θ_t defined in Eq. (8). Defining the per-period average consumption probability for an attacked item,

$$\omega(\delta) \;=\; \rho\, p(\delta,\alpha) + (1-\rho)\, p_0 \;\in\; [p_0,\, p_{\max}], \tag{12}$$

the active-attack expression becomes

$$\bar{c}_t \;=\; b\,\omega(\delta)\,\theta_t \;+\; b\,\rho\, p(\delta,\alpha)\,\delta, \qquad t \in \mathcal{T}_\pi. \tag{13}$$

Post-attack regime ($t > \tau$). After the attack, the perturbation is withdrawn, so every user observes the unmodified prediction $\tilde{P}^t_{ui} = P^t_{ui}$ and consumes attacked items at the baseline probability p_0. Substituting these into Eq. (10) (with $p(\delta,\alpha)$ replaced by p_0) merges the two user subgroups into a single population sum that, by the definition (8) of θ_t, equals $|\mathcal{I}_\pi|\,|\mathcal{U}|\,\theta_t$:

$$\bar{c}_t \;=\; \frac{b\,p_0}{|\mathcal{I}_\pi|\,|\mathcal{U}|} \sum_{i\in\mathcal{I}_\pi}\sum_{u\in\mathcal{U}} \left(P^t_{ui} - r_{ui}\right) \;=\; b\,p_0\,\theta_t, \qquad t > \tau. \tag{14}$$

Combining (13) and (14) yields the closed-form decomposition

$$\bar{c}_t \;=\; \begin{cases} b\,\omega(\delta)\,\theta_t \;+\; b\,\rho\, p(\delta,\alpha)\,\delta, & t \in \mathcal{T}_\pi \quad \text{(attack active)},\\ b\,p_0\,\theta_t, & t > \tau \quad \text{(attack over)}. \end{cases} \tag{15}$$

The two terms in (15) have distinct interpretations. The *bias-amplified injection* $b\,\rho\, p(\delta,\alpha)\,\delta$ is the new contamination introduced each period of attack: a fraction $\rho\, p(\delta,\alpha)$ of all user–item pairs consumes attacked items at the inflated rank, and each such consumption produces a rating biased $b\delta$ above the user's true preference. The *feedback term* $b\,\omega(\delta)\,\theta_t$ is the existing contamination that biased users echo back through their ratings: any current overprediction θ_t shifts new submitted ratings upward by $b\,\theta_t$, and this signal re-enters the RS through retraining. After the attack, the injection term vanishes and only the feedback term remains, weighted by the baseline consumption probability p_0.

A.4 Derivation of Contamination State θ_t and its Temporal Dynamics

Proposition 1. *Under the recurrence relation of temporal contamination dynamics (Equation 9), with $\theta_0 = 0$ and δ fixed,*

$$\theta_t \;=\; \theta^*(\delta)\big(1-\mu_\omega(b,\delta)^t\big), \quad t = 1,\dots,\tau, \qquad \theta_{\tau+k} \;=\; \mu_0(b)^k\,\theta_\tau, \quad k \ge 1, \tag{16}$$

where the perturbation-dependent fixed point is

$$\theta^*(\delta) \;=\; \frac{b\,\rho\, p(\delta,\alpha)\,\delta}{1 - b\,\omega(\delta)}, \tag{17}$$

and $\mu_\omega(b,\delta) = 1-\lambda\big(1-b\,\omega(\delta)\big)$ and $\mu_0(b) = 1-\lambda(1-b\,p_0)$ are the persistence factors of the perturbation within and after the attack window, respectively.[10]

Proof. Inside the attack window, substituting the active-attack branch of (15) into the recurrence

[10] They measure the fraction of current contamination that survives into the next period. Because $\omega(\delta) \ge p_0$, we have $\mu_\omega(b,\delta) \ge \mu_0(b)$: contamination decays *more slowly during* the attack than after.

equation (9) yields the affine recurrence

$$\theta_{t+1} \;=\; (1-\lambda)\,\theta_t \;+\; \lambda\big[b\,\omega(\delta)\,\theta_t + b\,\rho\,p(\delta,\alpha)\,\delta\big] \;=\; \mu_\omega(b,\delta)\,\theta_t \;+\; \lambda\,b\,\rho\,p(\delta,\alpha)\,\delta, \tag{18}$$

which has constant slope $\mu_\omega(b,\delta)$ and constant inhomogeneous term. Setting $\theta_{t+1} = \theta_t = \theta^*$ in (18) and solving gives the unique stationary point

$$\theta^*(\delta) \;=\; \frac{\lambda\,b\,\rho\,p(\delta,\alpha)\,\delta}{1-\mu_\omega(b,\delta)} \;=\; \frac{\lambda\,b\,\rho\,p(\delta,\alpha)\,\delta}{\lambda\big(1-b\,\omega(\delta)\big)} \;=\; \frac{b\,\rho\,p(\delta,\alpha)\,\delta}{1-b\,\omega(\delta)},$$

which is (17). Subtracting θ^* from both sides of (18) gives the homogeneous recurrence $\theta_{t+1} - \theta^* = \mu_\omega(b,\delta)(\theta_t - \theta^*)$, and iterating from the initial condition $\theta_0 = 0$ yields $\theta_t - \theta^* = -\mu_\omega(b,\delta)^t\,\theta^*$, i.e., $\theta_t = \theta^*(\delta)\big(1-\mu_\omega(b,\delta)^t\big)$ for $t = 1,\ldots,\tau$. After the attack, the injection term vanishes and substituting the post-attack branch of (15) into (9) produces the homogeneous recurrence $\theta_{t+1} = \mu_0(b)\,\theta_t$, whose iteration from θ_τ gives $\theta_{\tau+k} = \mu_0(b)^k\,\theta_\tau$ for $k \geq 1$. □

The fixed point $\theta^*(\delta)$ (expressed in Equation 17) is the equilibrium contamination at which contamination influx and the system's self-correction of contamination exactly balance. Its denominator $1 - b\,\omega(\delta)$ represents an *effective self-correction force*: if this quantity is large (i.e., when the preference bias and the consumption probability of the perturbed items are small), the system can self-correct (discount) the contamination at a higher rate.

Intuitively, Proposition 1 depicts the trajectory of the contamination state both within and after the attack window. During the attack, the rating inflation of perturbed items increases (i.e., the contamination state increases from 0 to $\theta^*(\delta)$ as time passes); after the attack, the system recovers, and the rating inflation of perturbed items decreases (i.e., it decays geometrically at the rate $\mu_0(b)$). Since $\mu_0(b)$ is strictly increasing in b, more biased populations recover more slowly.

A.5 The Self-Reinforcing Feedback Loop

Equations (16)–(17) encode three distinct mechanisms by which preference bias couples a transient external perturbation into durable RS distortion.

(i) Amplification during the attack. Each unit of injected signal δ generates equilibrium contamination $\theta^*(\delta)$, a monotonically increasing, convex function of bias b: the numerator $b\,\rho\,p\,\delta$ grows linearly in b while the denominator $1 - b\,\omega$ (the effective self-correction force) contracts simultaneously, amplifying the attack influence (i.e., the contamination) as the preference bias rises.

(ii) Accelerating growth during the attack. During the attack, the persistence factor $\mu_\omega(b,\delta)$ governs how quickly the contamination grows. With larger preference bias (i.e., $b \to 1$), the effective self-correction factor becomes smaller (i.e., $1 - b\,\omega(\delta) \to 0$). Consequently, the contamination persistence factor becomes larger (i.e., $\mu_\omega(b,\delta) \to 1$): the contamination grows more quickly over time (i.e., the speed of increasing contamination state becomes larger) when the persistence factor is large (due to a larger preference bias).[11]

(iii) Perpetuation after the attack. Once the external perturbation is withdrawn, biased users continue to anchor on the contaminated predictions and submit ratings $b(P^t_{ui} - r_{ui})$ above their true preferences (Eq. (7)). These ratings re-enter the RS through retraining and sustain the overpredictions. The post-attack persistence factor $\mu_0(b) = 1 - \lambda(1 - b\,p_0)$ stays significant as $b \to 1$, so a sufficiently biased population unwittingly maintains the distortion even long after the adversary withdraws.

[11] The contamination state during the attack is $\theta_t = \theta^*(\delta)\big(1-\mu_\omega(b,\delta)^t\big)$. The speed of increasing contamination can be obtained by taking the derivative of θ_t with respect to t, $\frac{\partial\theta_t}{\partial t} = \theta^*(\delta)(-\mu^t_\omega ln\mu_\omega)$. Notice that, by definition, $\theta^* \geq 0$ and $0 \leq \mu_\omega \leq 1$; and thus $\frac{\partial\theta_t}{\partial t}$ is a monotonically increasing, positive-valued function of μ_ω. This means that the speed of increasing contamination increases as the persistence factor μ_ω becomes larger (due to a larger preference bias).

Appendix B Hyper-Parameter Tuning for Recommendation Algorithms

Table 5 shows the exploration space for hyper-parameter tuning and the final hyper-parameter values (chosen using the grid search procedure). All experiments were conducted on a server with Nvidia A100 GPUs.

Table 5: Model Hyper-Parameters and Tuning Details

Algorithms	Hyper-Parameters	Final Choices (Netflix)	Final Choices (Yelp)	Grid Search Range
SVD	features	32	15	[10, 200]
	iterations	200	50	[10, 300]
	learning rate	0.001	0.001	[0.0001, 0.1]
	regularization	0.015	0.5	[0.001, 1]
	bias	True	True	True, False
ItemKNN	neighbors	50	25	[10, 200]
	minimum neighbors	5	7	[1, 20]
	minimum similarity	0.03	0.01	[0.001, 0.3]
	center	True	True	True, False
	aggregate	Weighted-average	Weighted-average	Weighted-average, Sum
LinUCB	feature type	stats	stats	stats, embedding
	regularization λ	1.0	10.0	stats: [0.01, 0.1, 1, 10, 100]; embedding: [0.01, 0.1, 1, 10]
	exploration coefficient	0.05	0.05	stats: [0.0, 0.05, 0.1, 0.5, 1.0]; embedding: [0.0, 0.05, 0.1]
	SVD latent factors k	N/A (stats)	N/A (stats)	[10, 20, 50] (embedding only)
	cross-validation folds	5	5	5 (fixed)
MHCL	embedding dimension	64	64	[32, 64]
	number of GNN layers	3	2	[1, 2, 3]
	number of hops	2	3	[1, 2, 3]
	contrastive temperature	0.2	0.1	[0.05, 0.1, 0.2]
	contrastive loss weight λ_{cl}	0.05	0.1	[0.01, 0.05, 0.1]
	learning rate	0.001	0.001	0.001 (fixed)
	optimizer	Adam	Adam	Adam (fixed)

LinUCB feature types. *stats*: A hand-crafted user feature vector computed from training ratings. Example features are: (1) the user's mean rating; (2) the deviation of the user's mean from the global mean; (3) a normalized log-count of the user's rating activity, etc. *embedding*: A user embedding derived from SVD of the user-item rating matrix.

Appendix C Experiment 1: Main Results for Four RS Algorithms and Two Datasets

This appendix presents the impact of preference bias and external perturbation on RS performance for different RS algorithms (SVD, ItemKNN, LinUCB, and MHCL) and rating datasets (Netflix and Yelp) in Figure 7 and 8.

Figure 7: Impact of Preference Bias and External Perturbation on Longitudinal RS Performance (Netflix)

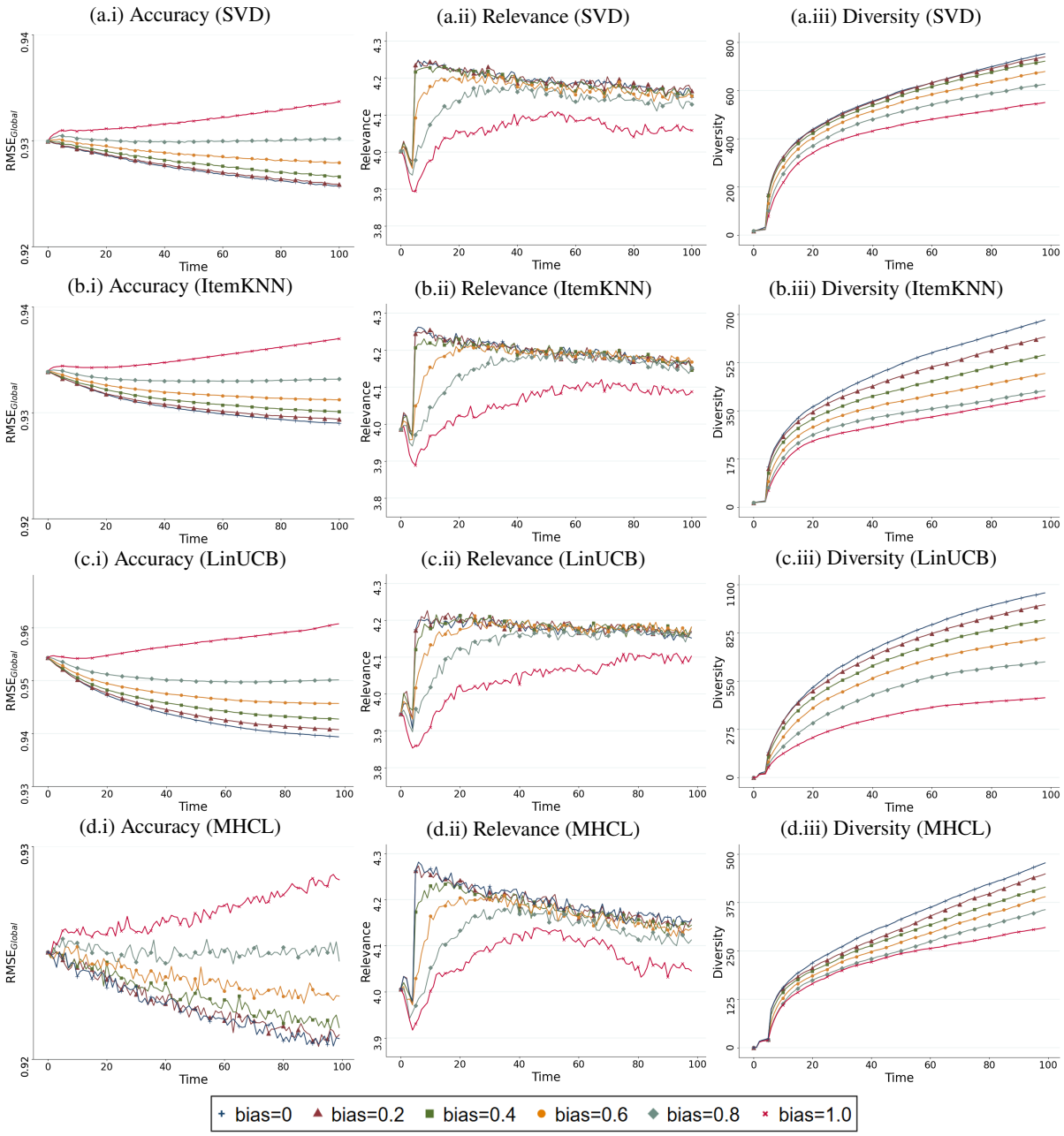
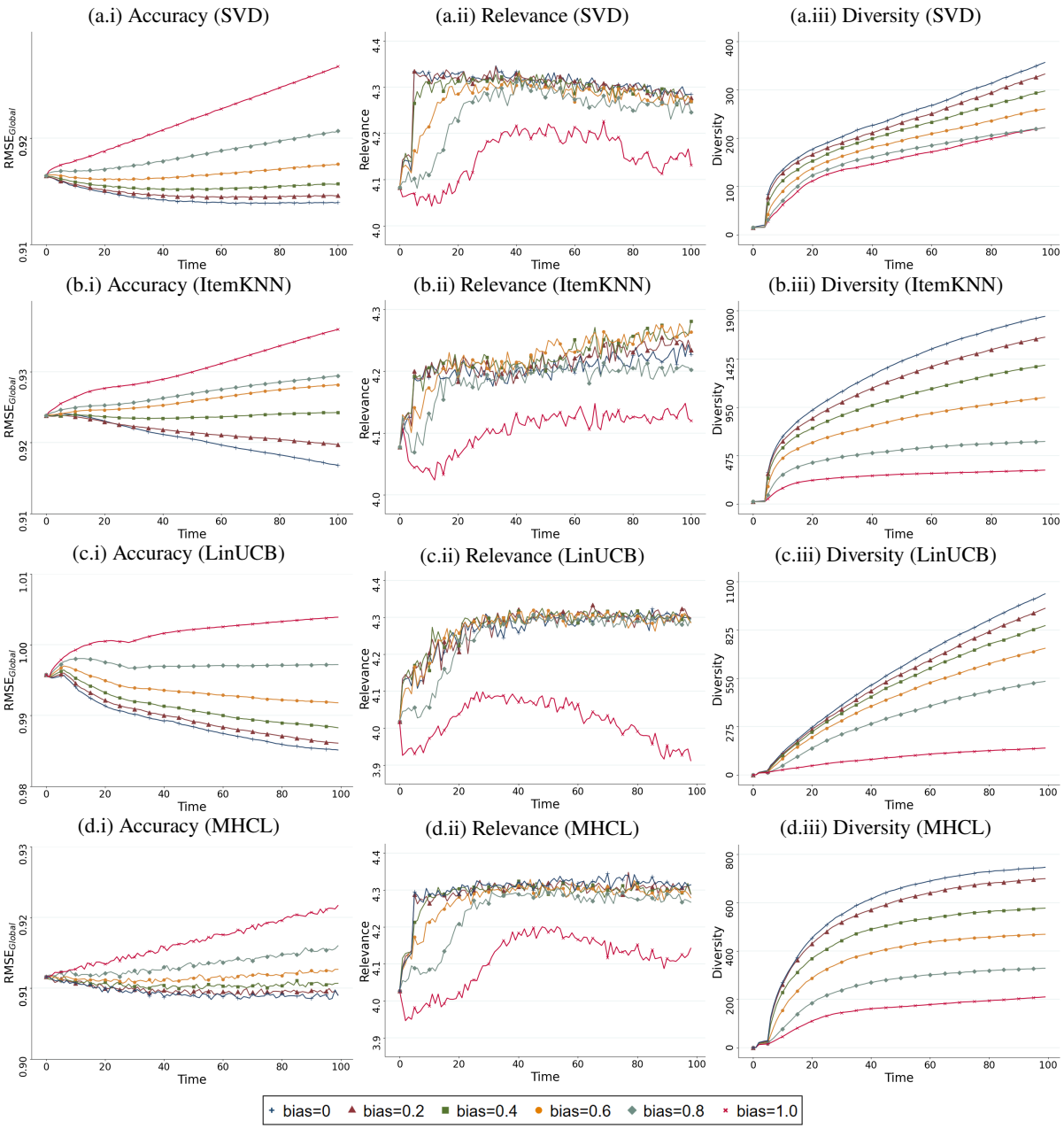


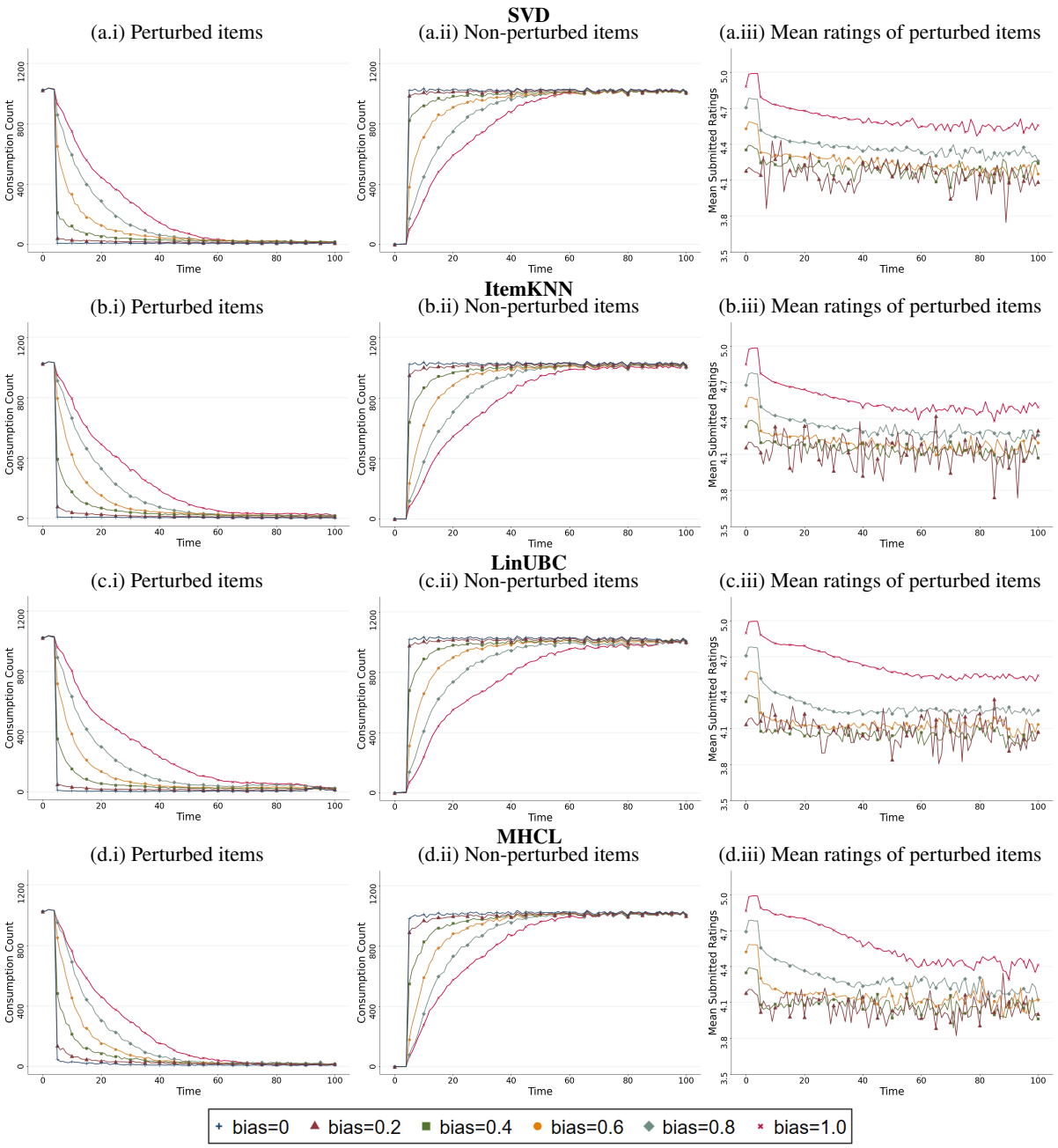
Figure 8: Impact of Preference Bias and External Perturbation on Longitudinal RS Performance (Yelp)



Appendix D Experiment 1: Consumption Analysis for Four RS Algorithms

This appendix provides the consumption analysis of perturbed and non-perturbed items for different RS algorithms (SVD, ItemKNN, LinUCB, MHCL) using Netflix data in Figure 9.

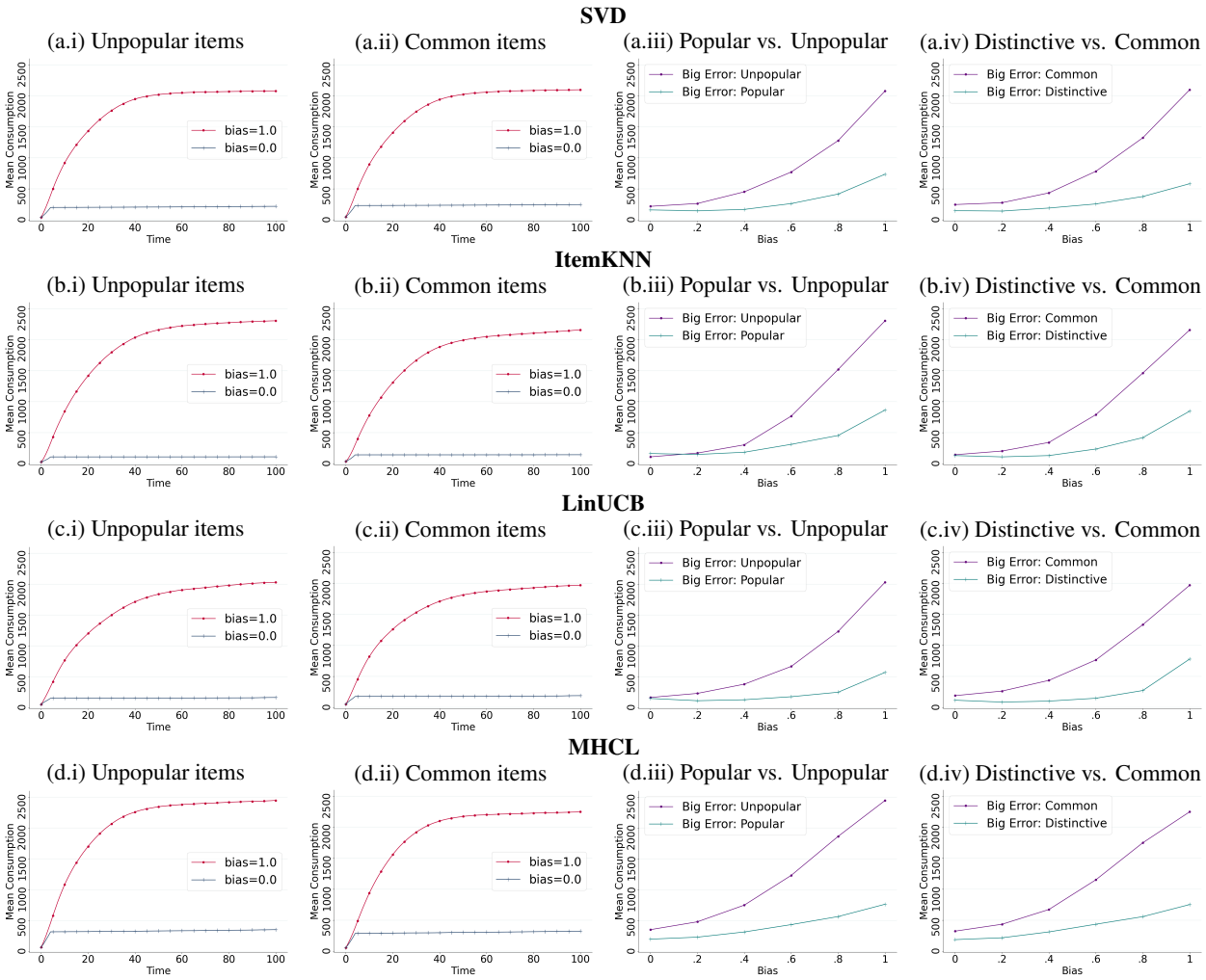
Figure 9: Consumption Analysis of Perturbed and Non-Perturbed Items Across Models (Netflix)



Appendix E Experiment 3: Big-Error Item Subgroup Analysis for Four RS Algorithms

This appendix provides the big-error item consumption analysis for different RS algorithms (SVD, ItemKNN, LinUCB, and MHCL) using Netflix data in Figure 10.

Figure 10: Mean Consumption of Big-Error Item Subgroups with Perturbations on All Users Across Models (Netflix)

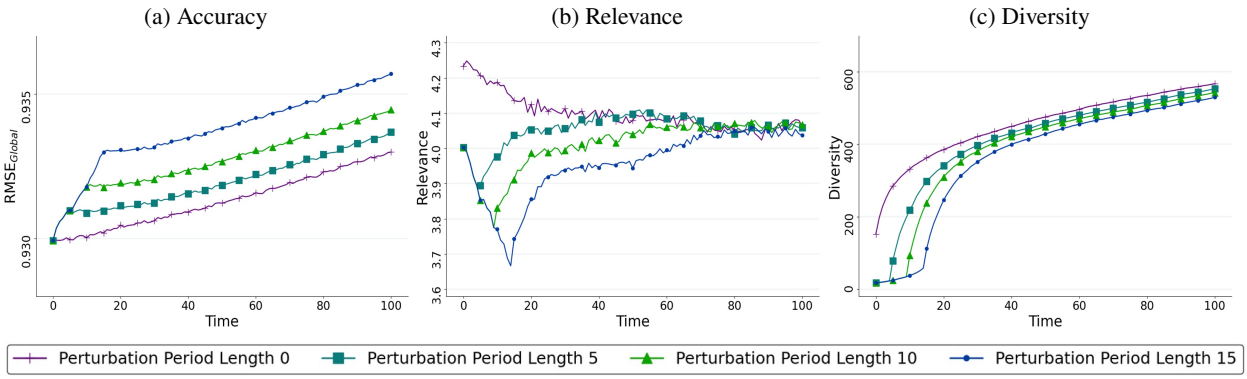


Appendix F Robustness Check: Varying Perturbation Period Length

We vary the length of the perturbation period as 0, 5, 10, and 15 time steps. We fix the preference bias level at 1.0 and keep all other parameters consistent with the main setup of Experiment 1 (Section 4). Figure 11 illustrates the effect of different perturbation lengths on RS performance and consumption outcomes. In general, RS predictive performance (RMSE) and consumption outcomes (diversity and relevance) are more adversely affected as the duration of the perturbation period increases. The larger adverse effect of the longer

external perturbation on RMSE lasts throughout the simulation period (Figure 11a). In contrast, the adverse effect on relevance and diversity dissipates after the perturbation period ends (Figures 11b and 11c). Notably, it takes significantly more time for the system to recover from a longer perturbation period; e.g., it takes about 35 timesteps for consumption relevance to recover to the normal level when the perturbation lasts 5 periods, but it takes more than 60 timesteps for it to recover when the system undergoes a 15-period perturbation (Figure 11b).

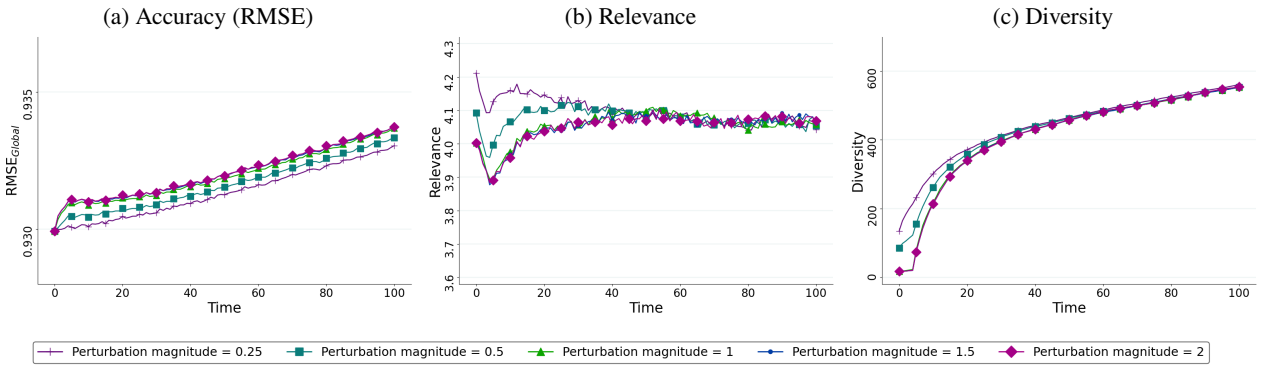
Figure 11: Varying Perturbation Period Length (Netflix, SVD)



Appendix G Robustness Check: Varying Perturbation Magnitude

We vary perturbation magnitude δ, i.e., the additive rating push applied by the attacker to each perturbed item, as 0.25, 0.5, 1, 1.5, and 2 on the 1–5 rating scale ($\delta = 1$ corresponds to the standard one-star perturbation used in the main setup). We fix the preference bias level at 1.0 and keep all other parameters consistent with the main setup of Experiment 1 (Section 4). Figure 12 illustrates the effect of different perturbation magnitudes on RS performance and consumption outcomes. As might be expected, the adverse effect of external perturbation grows with δ at small magnitudes: larger δ leads to a bigger and more persistent RMSE degradation throughout the simulation period (Figure 12a) and to a deeper drop in consumption relevance during the perturbation window (Figure 12b), while the post-perturbation recovery dynamics of relevance are similar across magnitudes. Consumption diversity is most heavily suppressed during the attack under larger δ, but eventually recovers to a comparable level across all magnitudes by the end of the simulation (Figure 12c). However, the attack saturates beyond $\delta = 1$: the trajectories for $\delta \in \{1, 1.5, 2\}$ are nearly indistinguishable on all three panels, indicating sharply diminishing returns to the attacker once δ exceeds about one star. The mechanism is straightforward: a one-star push is already large enough to lift the targeted items into the top-1 recommendation slot for essentially all biased users, so further increasing δ no longer changes which item is recommended (or consumed) and therefore does not amplify the attack's impact. Notably, even at the smallest tested magnitude ($\delta = 0.25$, i.e., a quarter-star push), the perturbation already produces clear distortions in all three outcomes, suggesting that under high preference bias even a modest rating push can leave a measurable, lasting imprint on RS predictive performance.

Figure 12: Varying Perturbation Magnitude (Netflix, SVD; preference bias = 1.0)



Appendix H Different Perturbation Strategies

In the main experiments, we focused on one specific form of small-scale external perturbation: increasing the system-predicted ratings of all perturbed items equally by one star. In practice, however, attackers may employ a variety of perturbation strategies. For example, the perturbation magnitude on each item may vary (e.g., an item may need less than a one-star rating adjustment to be "promoted" into the top-5 recommendations); or the perturbation may target items or users with certain characteristics (i.e., not at random)—e.g., less popular items may be intentionally targeted because they do not receive sufficient user attention. To examine whether our findings generalize across these alternative attack designs, this appendix compares six perturbation settings, defined in Table 6: **Setting 1**, the default setup that increases the system-predicted ratings of all perturbed items equally by one star; **Setting 2** ("Minimal Perturbation"), which uses the smallest necessary rating increase to promote the five highest-predicted perturbed items into the top-5 recommendation list; two variations of *popularity-based* perturbation (**Settings 3a** and **3b**), which select perturbed items from the top- and bottom-half of items ranked by popularity, respectively; and two variations of *consumption-frequency-based* perturbation (**Settings 4a** and **4b**), which apply perturbation only to top- and bottom-half frequent users, respectively. Across all settings, only 1% of all items are perturbed, the perturbation occurs only during the first five simulation periods, and we report the average values across ten simulation runs.

Table 6: Different Perturbation Settings

	Perturbed Item Selection	**Perturbed User Selection**	**Perturbation Approach**	**Total Consumption of Perturbed Items when** $bias = 1$	**Total Consumption of Perturbed Items when** $bias = 0$
Setting 1	Random 1% of all items	All users	Adding 1 star to the system-predicted rating	23,687	6,035
Setting 2	Random 1% of all items	All users	Promoting to top 5 by adding the minimum to the system-predicted rating	6,054	3,204
Setting 3a	Random 1% items, chosen from top-half popular items	All users	Adding 1 star to the system-predicted rating	22,722	7,004
Setting 3b	Random 1% items, chosen from bottom-half popular items	All users	Adding 1 star to the system-predicted rating	24,817	5,290
Setting 4a	Random 1% of all items	Top-half frequent users	Adding 1 star to the system-predicted rating	12,005	4,169
Setting 4b	Random 1% of all items	Bottom-half frequent users	Adding 1 star to the system-predicted rating	7,122	1,835

Comparing the default setting (Setting 1) with the minimal perturbation setting (Setting 2) in Figure 13, we

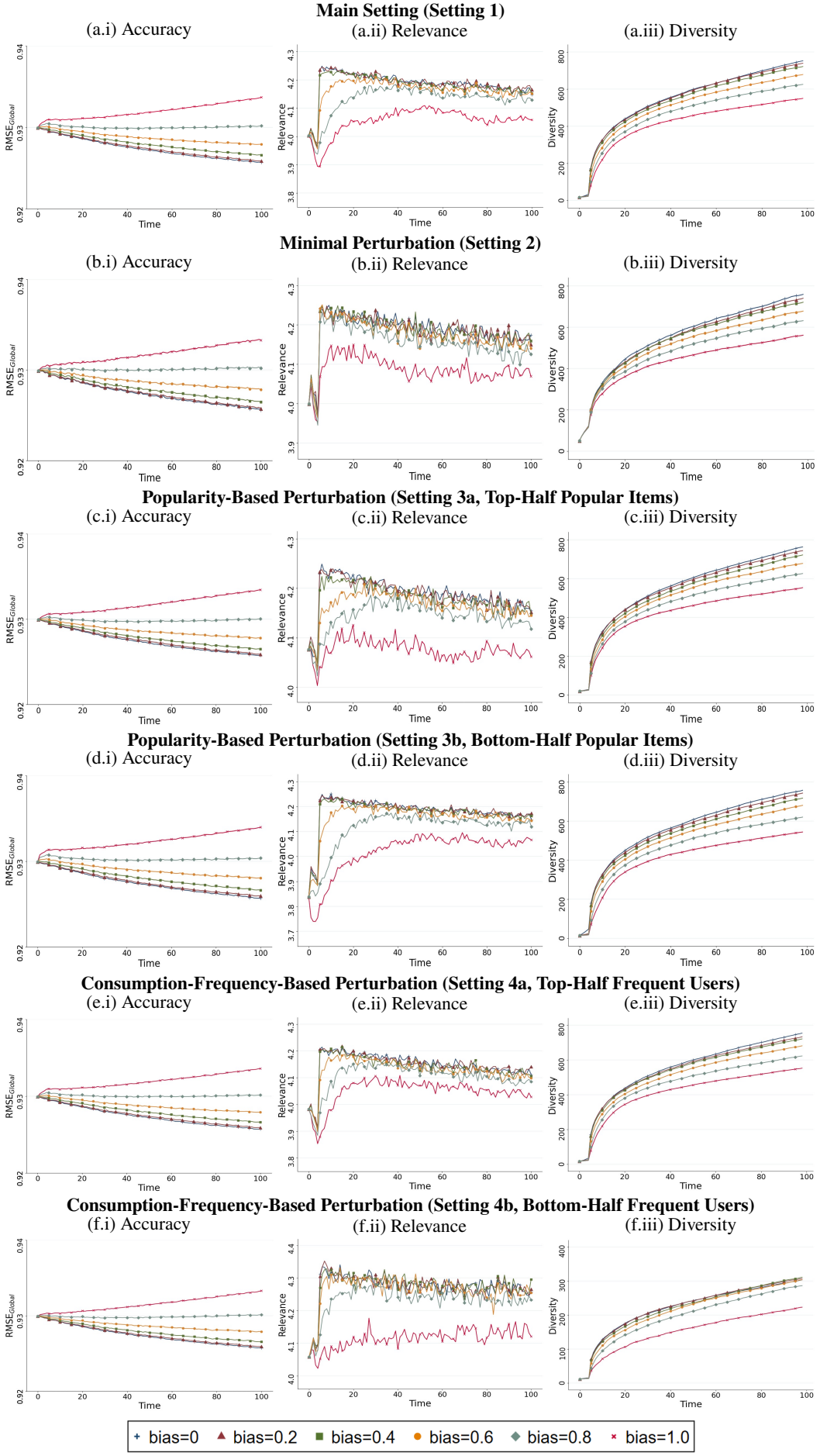
find that, with minimal perturbation, the consumption relevance drops less significantly, and it also recovers more quickly after the perturbation period. Intuitively, minimal perturbation injects less rating adjustment than the default 1-star perturbation approach. On average, the necessary additional rating adjustment in this setting is 0.5-0.8 stars. The average rating adjustment varies with the user's preference bias level. When preference bias is higher, smaller rating adjustments are needed because bias amplifies the perturbation effect; e.g., with bias level at 1.0, only 0.5 rating adjustment on average is needed to promote the top five perturbed items into top-5 recommendations. At the same time, under minimal perturbation, the system can more quickly correct the artificial rating inflation and reduce the lingering effects during the post-perturbation period.

Comparing the default perturbation (Setting 1) with the popularity-based perturbation (Settings 3a and 3b) in Figure 13, we find that perturbing less popular items (Setting 3b) leads to the most significant drop in consumption relevance during the perturbation period among all settings, and the system also recovers most slowly in the post-perturbation period (Figure 13(d.ii)). This is corroborated by the fact that the total consumption count of perturbed items in Setting 3b is markedly higher than that in Setting 1 or Setting 3a, when $bias = 1$ (Table 6). This provides yet additional evidence supporting the key insight from Experiment 3: less popular items are more susceptible to preference bias and consequently more instrumental in propagating bias through post-consumption feedback ratings, thus impacting the system's performance over time.

We further compare the default perturbation (Setting 1) with consumption-frequency-based perturbation (Settings 4a and 4b). We show the average system performance and consumption outcomes within the perturbed user segments, i.e., top- and bottom-half frequent users (Figures 13(e) and 13(f), respectively).[12] Specifically, when external perturbation only affects low-frequency users (Setting 4b), the adverse effects on the system performance and consumption outcomes are weaker during the perturbation period and linger for shorter periods after the perturbation ends. Intuitively, less frequent users submit fewer perturbed ratings because they consume less, so the system is less contaminated and recovers more quickly.

[12] Since we show average outcome within the perturbed user segment, consumption diversity of the bottom-half frequent users is much lower due to inherently lower frequency of consumption.

Figure 13: Impact of Different Perturbation Strategies on Recommender System Performance (Netflix, SVD)

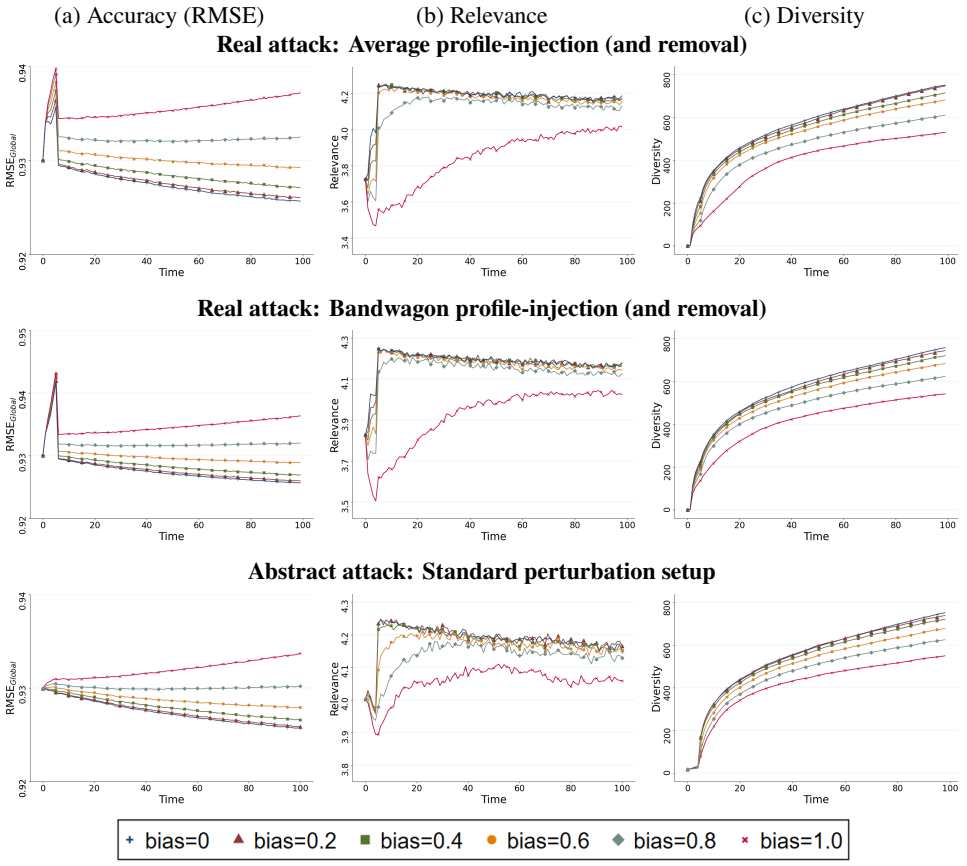


Appendix I Robustness Check: Real-World Attack vs. Abstract Attack

The main experiments rely on an abstract attack operationalization in which target items receive a fixed additive rating perturbation of $\delta = 1$ during the attack window, regardless of the underlying attack mechanism. To confirm that our findings are not specific to this formulation, we replace the abstract perturbation with two widely studied real-world profile-injection attacks: the *average attack*, in which sybil profiles assign the maximum rating to target items and draw filler ratings from each item's mean rating (mimicking realistic rating distributions), and the *bandwagon attack*, which additionally rates a fraction of high-popularity "head" items at the maximum to disguise sybils as enthusiastic real users (Mobasher et al. 2007, Si Li 2020). We configure the sybil profiles to be spread evenly over the same five-step injection window used in Experiment 1 (with sybil ratio = 0.20, target items = 1% of the catalog, 4% of the honest population injected per step, 2% filler item ratio[13]) and are then purged by an oracle defender at the end of the injection window, so that any post-attack distortion can be attributed to the residual contamination of honest users' submitted ratings rather than to the sybil profiles themselves. All six preference bias levels (0, 0.2, 0.4, 0.6, 0.8, and 1) are evaluated, and all other parameters are kept consistent with the main setup of Experiment 1 (Section 4). Figure 14 compares the average-attack setup (top row) and bandwagon-attack setup (middle row) against the abstract-perturbation standard setup (bottom row). The qualitative findings of Experiment 1 are representative with respect to both real-world attacks: during the injection window, RMSE first spikes sharply because of the contaminated training data and then drops back down after the oracle defender removes the sybils, but only the high-preference-bias trajectories continue to drift upward thereafter rather than returning to the no-bias baseline (Figure 14a); consumption relevance plunges deeply during the injection window with a slow, monotonically rising recovery whose pace is sharply slower at $bias = 1.0$ than at lower bias levels (Figure 14b); and consumption diversity is persistently suppressed under high bias throughout the post-attack period (Figure 14c). These results indicate that the bias-driven amplification and perpetuation of an external perturbation's adverse effects documented in the main experiments represent typical, realistic profile-injection attacks: regardless of the specific attack form, preference bias remains the main long-run driver of the system's inability to recover from a transient external perturbation.

[13]The filler items' ratings are drawn from a normal distribution with mean at the item's average rating and standard deviation at 0.3. For the average attack, the filler items are drawn randomly. For bandwagon attack, the filler items are drawn from the top 10% popular items.

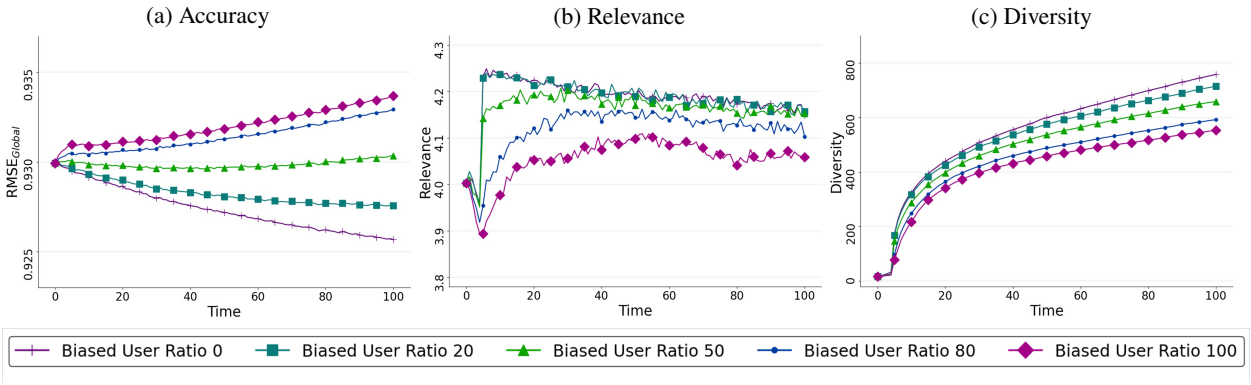
Figure 14: Real-Attack vs. Abstract-Attack Results (Netflix, SVD)



Appendix J Robustness Check: Varying Biased User Ratio

We simulate a heterogeneous user population with different portions of biased users: 20%, 50%, 80%, and 100%. We fix preference bias at 1.0 and hold all other parameters fixed as in Experiment 1 (Section 4). Figure 15 describes the effect of varying biased user ratio. As might be expected, RS accuracy and consumption outcomes are more impaired with larger portions of biased users. When more than half of the users exhibit preference bias, RS loses its ability to improve predictive performance with additional data (Figure 15a). In terms of consumption relevance, when biased users dominate the population (i.e., biased user ratio is 80% or 100%), RS cannot recover from the external perturbation as quickly and completely, as it is not able to neutralize the impact of perturbation even by the end of the simulation (Figure 15b). Finally, consumption diversity consistently and substantially decreases with larger biased user ratios (Figure 15c).

Figure 15: Varying Biased User Ratio (Netflix, SVD)



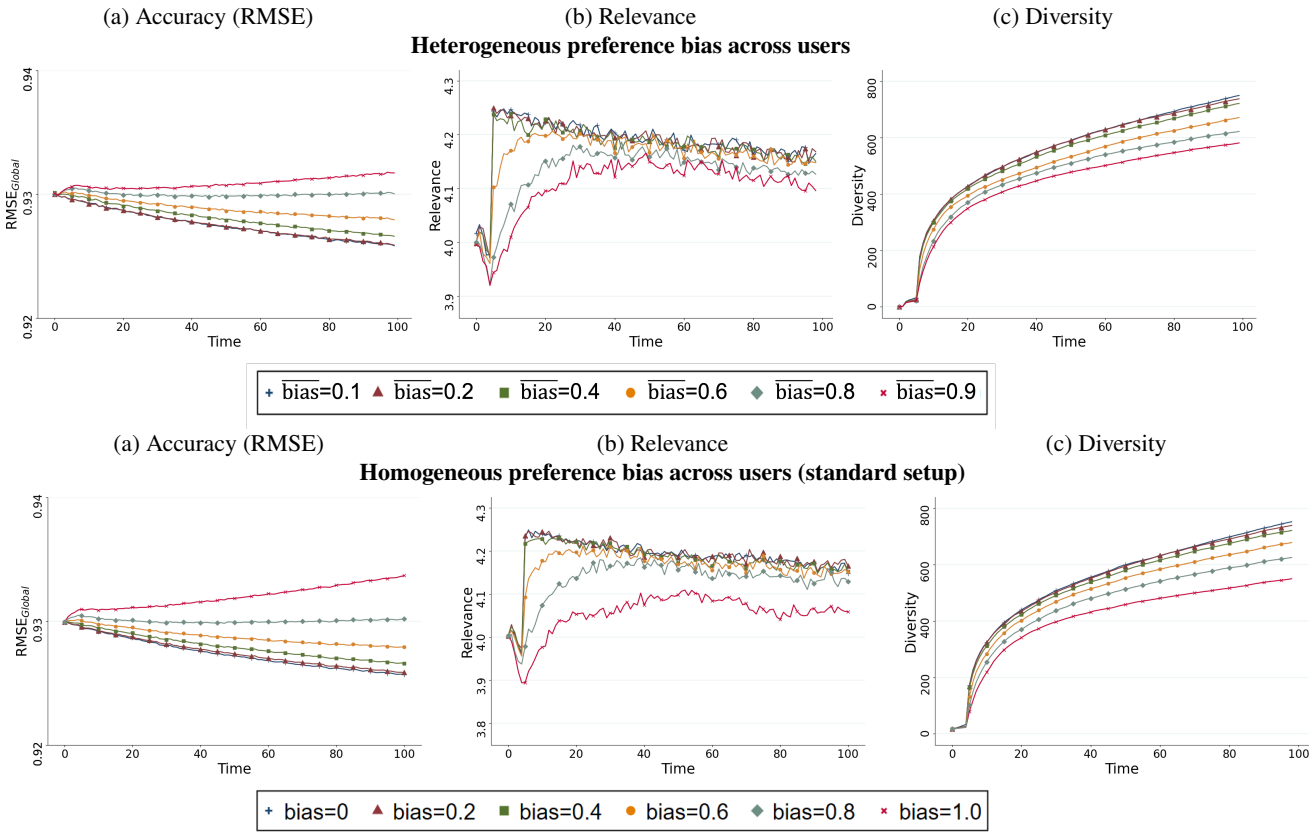
Appendix K Robustness Check: Heterogeneous User Preference Bias

In the main simulation experiments, all biased users share an identical preference-bias coefficient (homogeneous setup). To verify that our findings are not an artifact of this assumption, we instead draw each biased user's preference-bias coefficient $bias_u$ from a Beta distribution whose mean is set to the target bias level and whose standard deviation is fixed at 0.035 (so that, e.g., when the mean equals 0.1, approximately 99% of $bias_u$ values fall in $[0, 0.2]$). We vary the target bias level as 0.1, 0.2, 0.4, 0.6, 0.8, and 0.9 (avoiding the endpoints 0 and 1, since a non-degenerate Beta distribution on $(0, 1)$ cannot attain exact endpoint means), keeping all other parameters consistent with the main setup of Experiment 1 (Section 4). Figure 16 (on the next page) compares the heterogeneous setup (top row) against the homogeneous standard setup (bottom row). The qualitative findings of Experiment 1 are preserved: higher average bias still produces a larger RMSE degradation that persists throughout the simulation period (Figure 16a) and lower consumption diversity (Figure 16c).

Appendix L Robustness Check: Varying Reliance on Recommendations

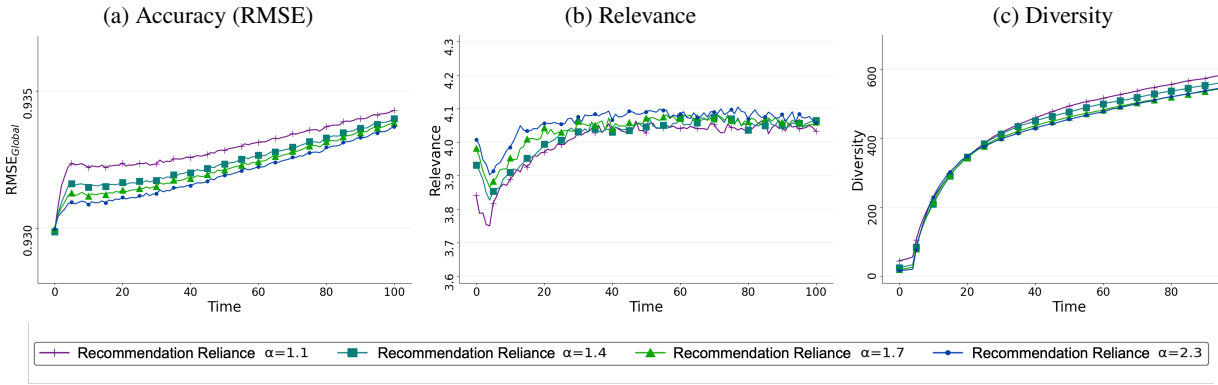
We vary the parameter α (i.e., the reliance on top recommendations), which controls how steeply users' selection probability decays with rank in the recommendation list (smaller α corresponds to more diffused consumption across the top of the list, while larger α concentrates consumption on the highest-ranked items), as 1.1, 1.4, 1.7, and 2.3. We fix the preference bias level at 1.0 and keep all other parameters consistent with

Figure 16: Robustness Check: Heterogeneous vs. Homogeneous Preference Bias Setup (Netflix, SVD)



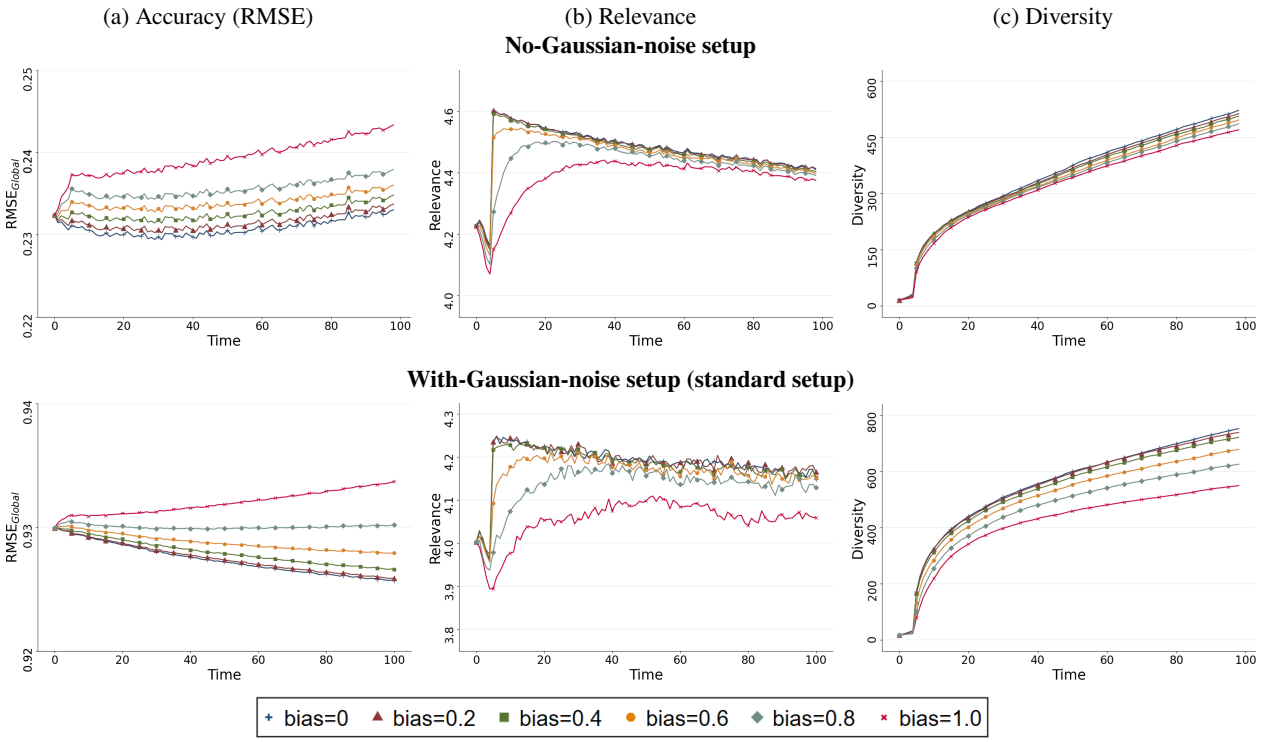
For the heterogeneous user-preference-bias setup, the boundary conditions use Beta means 0.1 and 0.9 (instead of exactly 0 and 1), because a non-degenerate Beta distribution on $(0, 1)$ cannot attain exact endpoint means; means 0 or 1 correspond only to degenerate point-mass cases.

Figure 17: Varying Reliance on Recommendations (Netflix, SVD; preference bias = 1)



the main setup of Experiment 1 (Section 4). Figure 17 (on the next page) illustrates the effect of different α on RS performance and consumption outcomes. The qualitative pattern of perturbation impact is preserved across all values of α, but the magnitude of the adverse effects systematically shrinks as α grows. With smaller α, RS exhibits a larger increase in RMSE that persists throughout the simulation period (Figure 17a) and a deeper drop in consumption relevance during the perturbation period (Figure 17b); consumption diversity is somewhat higher under smaller α (Figure 17c), because users sample more deeply down the list and therefore consume a wider variety of items overall. Overall, the main results and patterns are consistent across different values of α.

Figure 18: No-Gaussian-Noise vs. With-Gaussian-Noise Setup (Netflix, SVD)



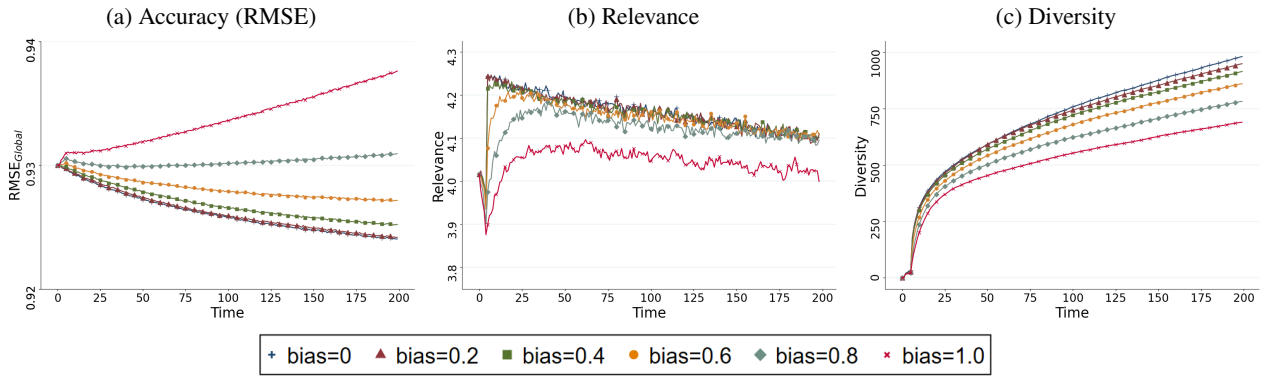
Appendix M User-Submitted Ratings without Gaussian Noise

In the main experiments, each user's ground-truth preference rating is generated from the underlying true preference plus an additive standard-normal noise term, $\mathcal{N}(0, 1)$, to mimic the idiosyncratic stochasticity in real-world rating behavior. To verify that our findings are not driven by this rating noise, we re-run Experiment 1 with the noise term removed. All six preference bias levels (0, 0.2, 0.4, 0.6, 0.8, and 1) are evaluated, and all other parameters are kept consistent with the main setup of Experiment 1 (Section 4). Figure 18 compares the no-Gaussian-noise setup (top row) against the standard with-Gaussian-noise setup (bottom row). The qualitative findings of Experiment 1 are preserved: higher preference bias still produces a larger and more persistent RMSE degradation throughout the simulation period (Figure 18a), a deeper drop in consumption relevance during the perturbation window with slower recovery thereafter (Figure 18b), and lower consumption diversity (Figure 18c). As might be expected, removing the rating noise lowers the overall RMSE level because the prediction target is intrinsically much easier to learn, but the bias-driven RMSE gap across preference bias levels and the relative ordering of the consumption-outcome trajectories remain unchanged.

Appendix N Robustness Check: Long Simulation Duration

To verify that the conclusions of Experiment 1 are not an artifact of the chosen simulation length, we extend the simulation horizon from $T = 100$ to $T = 200$ time steps while keeping all other parameters consistent with the main setup of Experiment 1 (Section 4). All six preference bias levels (0, 0.2, 0.4, 0.6, 0.8, and 1) are evaluated, and the external perturbation is still applied only during the first five periods. Figure 19 reports the resulting accuracy and consumption trajectories under the longer horizon. The qualitative findings from the main experiment carry over: higher preference bias produces a larger and more persistent RMSE gap that the RS does not close even after the additional 100 post-perturbation periods (Figure 19a); consumption

Figure 19: Long Simulation Duration (Netflix, SVD)



relevance under high bias remains depressed below the no-bias baseline well beyond the original $T = 100$ window (Figure 19b); and consumption diversity under high bias continues to lag the no-bias case throughout the extended simulation (Figure 19c).

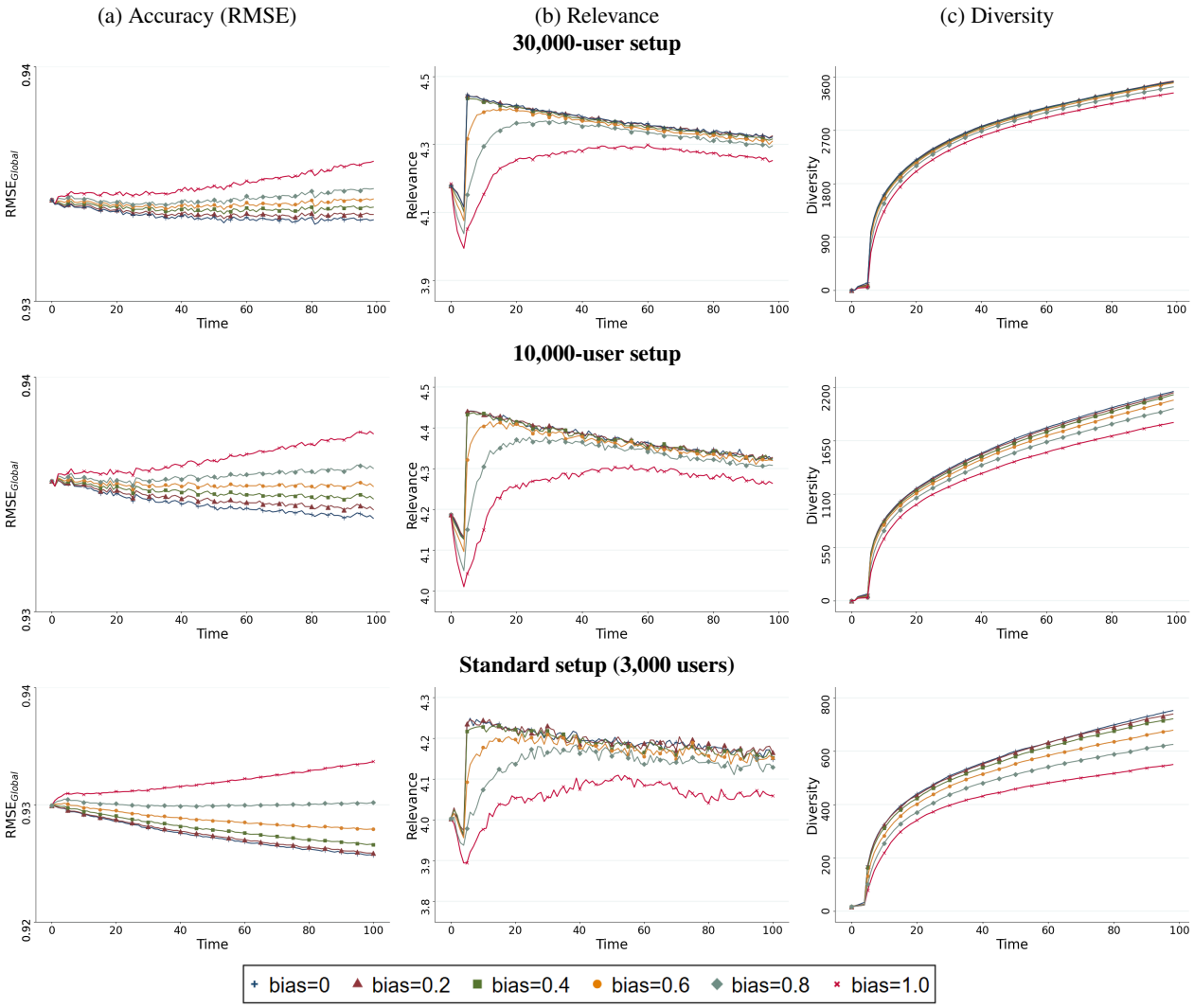
Appendix O Robustness Check: 10k- and 30k-User Seeding Datasets

The main experiments use a small-scale simulation seeded with 3,000 users; we now verify that our findings carry over to larger seeding datasets containing 10,000 and 30,000 Netflix users. All six preference bias levels (0, 0.2, 0.4, 0.6, 0.8, and 1) are evaluated, and all other parameters are kept consistent with the main setup of Experiment 1 (Section 4). Figure 20 compares the 30,000-user setup (top row), the 10,000-user setup (middle row), and the 3,000-user baseline (bottom row). The qualitative findings of Experiment 1 are preserved at all three scales: higher preference bias still produces a larger and more persistent RMSE rise (Figure 20a), a deeper drop in consumption relevance with slower post-attack recovery (Figure 20b), and persistently lower consumption diversity (Figure 20c). As the seeding dataset grows, the absolute spread of RMSE across preference-bias levels narrows modestly because each newly submitted rating contributes a smaller share of the overall training signal, but the relative ordering of the bias levels and the bias-driven amplification of the perturbation's adverse effects remain unchanged.

Appendix P Statistical Tests for Differences Across Bias Levels

This appendix provides formal statistical support for the outcome differences across bias levels observed in the longitudinal results of Experiment 1. For each combination of the recommendation algorithm (SVD, ItemKNN, LinUCB, and MHCL) and dataset (Netflix in Table 7; Yelp in Table 8), we conduct one-sided paired t-tests that compare each non-zero preference-bias level $bias > 0$ to the zero-bias baseline, using the ten matched simulation replications as paired observations. The tests are applied separately to accuracy (RMSE, hypothesized to increase under bias), consumption relevance (hypothesized to decrease), and consumption diversity (hypothesized to decrease), and are evaluated at two representative timesteps — one soon after the perturbation window (at $t = 15$, representing short-term effects) and one near the end of the simulation horizon (at $t = 100$, representing long-term effects). Across both datasets and all four algorithms, the results corroborate the qualitative claims of Section 4: bias-driven degradation in consumption diversity is statistically significant ($p < 0.05$) at essentially every bias level $bias \geq 0.2$, and when $t = 15$, accuracy show significant degradation from moderate bias levels onward ($bias \geq 0.4$), with significance levels strengthening monotonically in $bias$. Most importantly for our argument about persistence, nearly every effect that is statistically significant at $t = 15$ remains so at $t = 100$ (especially for the accuracy and diversity metrics),

Figure 20: Scale Comparison: 30k-User, 10k-User, and 3k-User Setup (Netflix, SVD)



providing formal evidence that preference bias does not merely amplify the perturbation's contemporaneous impact but also delays the system's recovery long after the perturbation window has closed.

Table 7: Timestep-Specific One-Sided Paired t-Tests: Accuracy, Consumption Relevance, and Diversity (Netflix Dataset)

	$t = 15$			$t = 100$		
bias	Acc.	Rel.	Div.	Acc.	Rel.	Div.
Algorithm: SVD						
$bias = 0.0$ (ref.)	0.9286	4.2210	430.8	0.9257	4.1476	757.6
$bias = 0.2$	0.9287^{**}	4.2213	426.9	0.9259^{***}	4.1651	743.9^{***}
$bias = 0.4$	0.9290^{***}	4.2173	415.1^{**}	0.9266^{***}	4.1521	725.9^{***}
$bias = 0.6$	0.9295^{***}	4.1964^{***}	394.8^{***}	0.9279^{***}	4.1514	681.6^{***}
$bias = 0.8$	0.9300^{***}	4.1407^{***}	363.0^{***}	0.9302^{***}	4.1290	628.3^{***}
$bias = 1.0$	0.9310^{***}	4.0585^{***}	332.6^{***}	0.9337^{***}	4.0588^{***}	553.5^{***}
Algorithm: ItemKNN						
$bias = 0.0$ (ref.)	0.9318	4.2449	357.4	0.9290	4.1486	686.7
$bias = 0.2$	0.9319	4.2366	339.4^{***}	0.9294^{***}	4.1525	622.8^{***}
$bias = 0.4$	0.9323^{***}	4.2358	313.8^{***}	0.9301^{***}	4.1451	557.6^{***}
$bias = 0.6$	0.9326^{***}	4.2092^{***}	285.2^{***}	0.9312^{***}	4.1675	488.5^{***}
$bias = 0.8$	0.9333^{***}	4.1305^{***}	257.5^{***}	0.9332^{***}	4.1519	426.9^{***}
$bias = 1.0$	0.9343^{***}	4.0178^{***}	235.5^{***}	0.9370^{***}	4.0880^{***}	406.5^{***}
Algorithm: LinUCB						
$bias = 0.0$ (ref.)	0.9475	4.2071	501.7	0.9393	4.1524	1062.4
$bias = 0.2$	0.9478^{***}	4.1946	483.9^{**}	0.9407^{***}	4.1524	992.2^{***}
$bias = 0.4$	0.9485^{***}	4.1976	441.8^{***}	0.9427^{***}	4.1502	908.0^{***}
$bias = 0.6$	0.9496^{***}	4.1626^{**}	384.0^{***}	0.9457^{***}	4.1608	802.1^{***}
$bias = 0.8$	0.9513^{***}	4.1252^{***}	300.9^{***}	0.9502^{***}	4.1483	664.8^{***}
$bias = 1.0$	0.9547^{***}	4.0182^{***}	220.8^{***}	0.9610^{***}	4.1022^{**}	459.0^{***}
Algorithm: MHCL						
$bias = 0.0$ (ref.)	0.9234	4.2368	219.2	0.9209	4.1529	482.0
$bias = 0.2$	0.9238	4.2381	205.7^{**}	0.9208	4.1304^{*}	453.4^{**}
$bias = 0.4$	0.9245^{*}	4.2330	197.9^{***}	0.9213	4.1222^{**}	419.4^{***}
$bias = 0.6$	0.9245^{*}	4.1939^{***}	185.8^{***}	0.9227^{***}	4.1350	394.1^{***}
$bias = 0.8$	0.9250^{***}	4.1197^{***}	175.0^{***}	0.9252^{***}	4.1000^{*}	360.3^{***}
$bias = 1.0$	0.9258^{***}	4.0504^{***}	165.7^{***}	0.9287^{***}	4.0397^{***}	313.2^{***}

Metrics are evaluated at two specific raw timesteps: $t = 15$ and $t = 100$. *Accuracy* (RMSE): full-test-set RMSE from the pre-computed log files at the given timestep. *Consumption Relevance*: mean true preference rating of consumed items at the given timestep. *Diversity*: cumulative unique-item count up to the given timestep. One-sided paired t-tests (10 matched pairs) compare each $bias > 0$ against $bias = 0$. RMSE is expected to increase; Relevance and Diversity to decrease. Stars on the treatment mean. $^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$. Among all the statistically significant differences in the table, 96% of them have an effect size (Cohen's d) that is considered large (>0.8).

Table 8: Timestep-Specific One-Sided Paired t-Tests: Accuracy, Consumption Relevance, and Diversity (Yelp Dataset)

	$t = 15$			$t = 100$		
bias	Acc.	Rel.	Div.	Acc.	Rel.	Div.
Algorithm: SVD						
$bias = 0.0$ (ref.)	0.9150	4.3276	174.3	0.9139	4.2849	361.4
$bias = 0.2$	0.9152^{**}	4.3206	163.1^{**}	0.9146^{***}	4.2772	336.2^{***}
$bias = 0.4$	0.9156^{***}	4.3054	150.0^{***}	0.9157^{***}	4.2690	300.9^{***}
$bias = 0.6$	0.9162^{***}	4.2833^{*}	133.8^{***}	0.9176^{***}	4.2698	264.1^{***}
$bias = 0.8$	0.9170^{***}	4.2354^{***}	119.5^{***}	0.9207^{***}	4.2458^{**}	223.2^{***}
$bias = 1.0$	0.9187^{***}	4.0748^{***}	108.4^{***}	0.9267^{***}	4.1318^{***}	224.2^{***}
Algorithm: ItemKNN						
$bias = 0.0$ (ref.)	0.9232	4.2025	901.7	0.9168	4.2267	1855.0
$bias = 0.2$	0.9230	4.2216	823.3^{***}	0.9197^{***}	4.2339	1648.8^{***}
$bias = 0.4$	0.9237^{***}	4.2271	730.7^{***}	0.9243^{***}	4.2809	1372.1^{***}
$bias = 0.6$	0.9246^{***}	4.2128	593.2^{***}	0.9281^{***}	4.2635	1051.8^{***}
$bias = 0.8$	0.9252^{***}	4.1791	398.8^{***}	0.9294^{***}	4.2022^{*}	614.7^{***}
$bias = 1.0$	0.9276^{***}	4.0737^{***}	229.0^{***}	0.9360^{***}	4.1197^{***}	334.7^{***}
Algorithm: LinUCB						
$bias = 0.0$ (ref.)	0.9914	4.2385	271.5	0.9850	4.2885	1044.6
$bias = 0.2$	0.9921^{***}	4.2337	258.4^{**}	0.9860^{***}	4.2888	960.8^{***}
$bias = 0.4$	0.9932^{***}	4.2523	244.4^{***}	0.9882^{***}	4.2875	861.8^{***}
$bias = 0.6$	0.9949^{***}	4.2677	214.8^{***}	0.9918^{***}	4.2723	729.2^{***}
$bias = 0.8$	0.9975^{***}	4.2114	152.1^{***}	0.9972^{***}	4.2832	537.7^{***}
$bias = 1.0$	1.0005^{***}	4.0312^{***}	52.6^{***}	1.0039^{***}	3.9207^{***}	153.7^{***}
Algorithm: MHCL						
$bias = 0.0$ (ref.)	0.9101	4.3000	453.3	0.9091	4.3095	748.5
$bias = 0.2$	0.9100	4.2830	431.4^{*}	0.9100^{***}	4.3099	703.1
$bias = 0.4$	0.9109^{*}	4.3017	371.0^{***}	0.9103^{**}	4.3106	580.6^{***}
$bias = 0.6$	0.9112^{**}	4.2743^{*}	290.0^{***}	0.9132^{***}	4.2819	473.3^{***}
$bias = 0.8$	0.9121^{***}	4.2029^{***}	188.4^{***}	0.9156^{***}	4.2512^{**}	333.1^{***}
$bias = 1.0$	0.9134^{***}	4.0063^{***}	114.7^{***}	0.9220^{***}	4.1191^{***}	215.9^{***}

Metrics are evaluated at two specific raw timesteps: $t = 15$ and $t = 100$. *Accuracy* (RMSE): full-test-set RMSE from the pre-computed log files at the given timestep. *Consumption Relevance*: mean true preference rating of consumed items at the given timestep. *Diversity*: cumulative unique-item count up to the given timestep. One-sided paired t-tests (10 matched pairs) compare each $bias > 0$ against $bias = 0$. RMSE is expected to increase; Relevance and Diversity to decrease. Stars on the treatment mean. $^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$. Among all the statistically significant differences in the table, 95% of them have an effect size (Cohen's d) that is considered large (>0.8).

Appendix Q Main Experiment Results with Standard Deviations

This appendix complements the mean-trajectory plots in Section 4 by adding the run-to-run variability of the underlying simulation replications. We re-display the longitudinal accuracy, consumption-relevance, and consumption-diversity trajectories of Experiment 1 for each combination of the recommendation algorithm (SVD, ItemKNN, LinUCB, and MHCL) and dataset (Figure 21 for Netflix; Figure 22 for Yelp), now with shaded ± 1 standard-deviation ribbons computed across the ten simulation replications. Across most algorithm-dataset combinations, the ribbons are visibly narrow relative to the spacing between adjacent preference-bias curves — and the contrast is particularly stark in the post-perturbation regime, where the cross-bias separation in accuracy and diversity is substantial. The bias-driven degradation patterns reported in the main text therefore reflect systematic effects of preference bias, and together with the paired t-tests in Appendix P they constitute two complementary forms of evidence that the cross-bias differences documented in Experiment 1 are statistically robust.

Figure 21: Impact of Preference Bias and External Perturbation on Longitudinal RS Performance (Netflix)

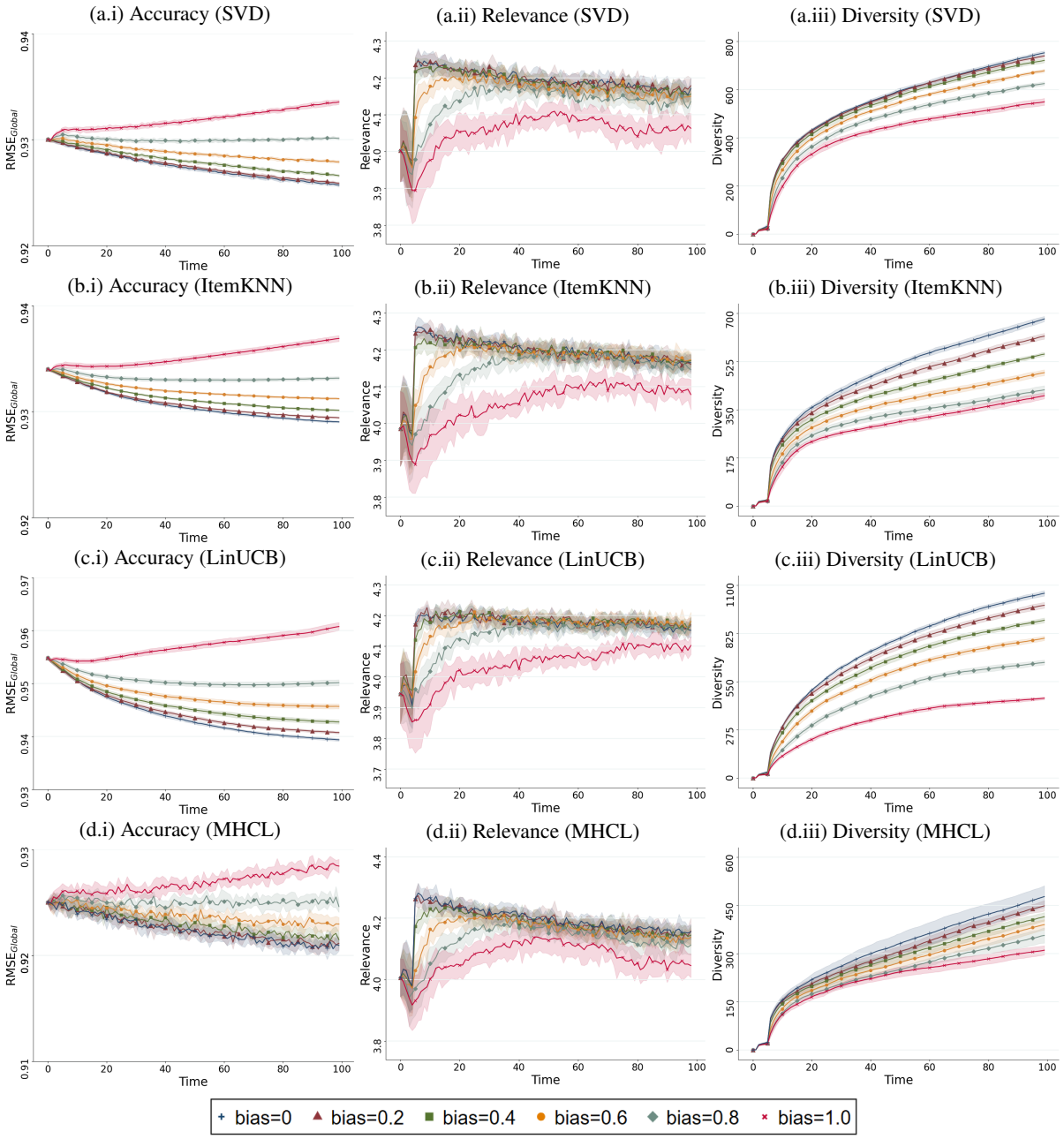


Figure 22: Impact of Preference Bias and External Perturbation on Longitudinal RS Performance (Yelp)