

AlignPoison: Conditional Targeted Poisoning of Generative Recommenders via Decoding-Aligned Prefixes

Zhuoxuan Zhang^{1,*}, Jinhao Duan^{1,*}, Xinyu Zhao¹, Jinduo Guo², Shuai Shao³, Rui Li³, Luke Simon³, Zhun Deng³, Tianlong Chen¹, Bo Sun³

¹University of North Carolina at Chapel Hill, ²University of Notre Dame, ³Meta

*Equal contribution.

Generative recommendation models based on autoregressive decoding have been adopted as alternatives to classical ranking-based systems. While their modeling flexibility has been widely studied, their resilience to data poisoning remains largely unexplored. In this work, we ask: how susceptible are generative recommenders to conditional targeted poisoning? Under a realistic setting where an adversary can inject limited malicious interactions and query a recommendation API as a black box, our empirical study shows that generative recommenders are vulnerable: injecting fewer than 1% synthetic interactions significantly increases the likelihood of promoting malicious targets under the trigger. To characterize this vulnerability, we analyze the synthesis of malicious interactions from two perspectives: (i) decoding alignment and (ii) prefix popularity. We observe that short, context-aligned prefixes that are both common under the model’s induced transition dynamics and strongly linked to a frequent trigger consistently amplify targeted promotion. Motivated by this analysis, we develop AlignPoison, a black-box framework that discovers and samples context-aligned prefixes from API outputs. We conduct experiments on benchmarks in unimodal and multimodal generative recommendation settings. Results show that these prefixes outperform competitive baselines, improving ASR lift by up to 16.3 percentage points and highlighting a decoding-dependent, context-sensitive poisoning vulnerability in generative recommendation models.

Date: October 10, 2026



1 Introduction

The recommendation paradigm is undergoing a fundamental shift from multi-stage ranking pipelines toward end-to-end unified generative architectures (Zhao et al., 2026). By combining user histories and multimodal signals into a single generative task, these systems consistently outperform conventional multi-stage methods (Rajput et al., 2023; Zhou et al., 2025a,b). This strong empirical performance has accelerated their transition from academic prototypes to large-scale industrial deployment, exemplified by systems such as OneRec on major short-video platforms (Zhou et al., 2025a; Liu et al., 2026). At the same time, this architectural shift introduces a different attack surface: rather than scoring a fixed candidate set or relying on natural-language item descriptions, generative recommenders autoregressively decode semantic item IDs, as illustrated in Figure 1. Notably, this distinction raises an immediate robustness question. Despite their rapid adoption, it remains unclear whether this shift inherits, mitigates, or amplifies the data poisoning risks documented in traditional sequential systems (Song et al., 2020; Zhang et al., 2024; Zhao et al., 2025).

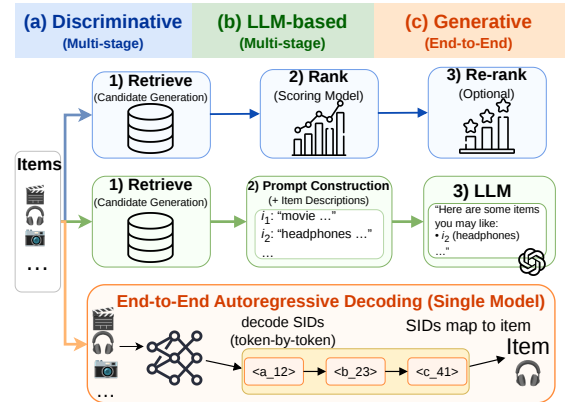


Figure 1 Conventional ((a)–(b)) vs. generative ((c)) recommendation architectures.

Poisoning attacks manipulate exposure and downstream user behavior, threatening platform integrity, safety, and fairness (Mobasher et al., 2007; Fang et al., 2018, 2020; Song et al., 2020). By injecting a small number of crafted user-item interactions into training data, attackers steer recommendations toward a target item in specific user-history contexts (Tang et al., 2020; Fang et al., 2020; Song et al., 2020; Thanh et al., 2022; Ning et al., 2026). We study conditional targeted poisoning of end-to-end generative recommenders with limited interaction injection and black-box API access.

RQ1 (Susceptibility): Are end-to-end generative recommenders vulnerable to conditional targeted poisoning at small poisoning ratios? Even simple trigger-conditioned injections substantially increase target exposure (Section 3.4).

RQ2 (Mechanism): What properties of synthetic interaction histories (malicious prefixes) determine attack effectiveness in generative recommenders? We identify two key properties: *decoding alignment* and *prefix popularity* (Section 4).

Based on these insights, we propose AlignPoison, which discovers context-aligned prefixes through API queries and injects synthetic histories to reinforce trigger→target associations. Across two Amazon product datasets and a multimodal micro-video dataset, AlignPoison outperforms poisoning baselines while preserving benign recommendation quality.

To our knowledge, attacks of end-to-end generative recommenders remains under-explored. Prior studies primarily address ranking-based, representation-based, or LLM-based recommenders (Mobasher et al., 2007; Fang et al., 2018, 2020; Song et al., 2020; Tang et al., 2020; Thanh et al., 2022; Ning et al., 2024; Wang et al., 2024b). These systems score candidates, match embeddings, or use natural-language inputs and item descriptions (Wang et al., 2025). Our setting instead concerns direct decoding of structured item tokens, motivating a decoding-aware study of how malicious histories affect generation. Our contributions are:

- **Empirical vulnerability study:** We demonstrate conditional targeted promotion using simple synthetic histories at small poisoning ratios (RQ1).
- **Mechanistic analysis:** We identify two key dimensions of malicious prefixes, *decoding alignment* and *prefix popularity*, and characterize how their interaction drives attack success (RQ2).
- **Black-box framework:** We develop a black-box prefix discovery, optimization, and adaptive sampling framework that recovers short, context-aligned prefixes using only API outputs.
- **Extensive evaluation:** We validate on two Amazon Review subsets (*Office_Products* and *Industrial_and_Scientific*) and the micro-video dataset MicroLens-100K, with comparisons to strong baselines, defenses, and ablations.

2 Related Work

2.1 Generative Recommendation

Generative recommendation reframes retrieval as sequence generation, with models decoding item identifiers rather than matching embeddings. Early work such as TIGER (Rajput et al., 2023) introduced encoder-decoder retrieval with semantic IDs, while subsequent systems extended this paradigm to industrial and multimodal settings through stronger architectures, alternative decoding schemes, and richer tokenization strategies, including OneRec, MTGR, NAR4Rec, MDGR, and related semantic-ID methods (Deng et al., 2025; Han et al., 2025; Ren et al., 2024; Shah et al., 2025; Feng et al., 2022; Wang et al., 2024a; Wei et al., 2025; Geng et al., 2023; Luo et al., 2025; Tan et al., 2024; Yang et al., 2025; Zhou et al., 2025b). A parallel line integrates language interfaces into recommendation, for example through unified prompting or lightweight LLM adaptation (Geng et al., 2022; Zheng et al., 2024). We study conditional item-promotion attacks on these structured-token recommenders.

2.2 Data Poisoning Attacks in Recommendation and Autoregressive Models

Prior work explores recommender poisoning (Mobasher et al., 2007; Song et al., 2020; Zhao et al., 2025; Wang et al., 2024b; Li et al., 2025), adversarial prompting and poisoning in language models (Zou et al., 2023; Wan et al., 2023; Souly et al., 2025), and text or profile manipulation in language-based recommendation (Wang et al., 2025; Ning et al., 2026, 2024). We study semantic-ID generative recommenders under top- K query access and interaction-history injection, with metadata, prompts, and the tokenizer fixed. These constraints preclude direct text manipulation and motivate decoding-aligned interaction poisoning, evaluated against adapted interaction-based baselines.

3 Preliminaries

3.1 Generative Recommendation Model

We consider *generative recommender systems* that recommend items by generating their semantic identifiers (SIDs) autoregressively (Rajput et al., 2023; Zhou et al., 2025b). Let $\mathcal{I}$ denote the item universe, and let $\tau : \mathcal{I} \rightarrow \mathcal{V}^*$ map each item $i \in \mathcal{I}$ to a SID token sequence over vocabulary $\mathcal{V}$. Given a user history $H = (i_1, \dots, i_m)$, the model serializes it as $x = \tau(i_1) \oplus \dots \oplus \tau(i_m)$ and uses a decoder parameterized by θ to generate the next item’s SID one token at a time. At inference time, beam search is used to decode high-probability SID sequences; each decoded SID sequence is then mapped back to its original item, and the resulting top- K items form $\text{Rec}_K(H)$.

3.2 Data Poisoning Attacks

We discuss item-promotion attacks carried out through data poisoning in generative recommenders. Let $\mathcal{D} = \{H_j\}_{j=1}^N$ be the clean set of user histories and let $\mathcal{D}_{\text{poison}}$ be the injected synthetic histories. The poisoned training set is $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}_{\text{poison}}$, and the poisoning ratio is $r = |\mathcal{D}_{\text{poison}}|/|\mathcal{D}'|$. Poisoning can be untargeted, aiming to degrade overall quality, or targeted, aiming to promote a specific item. We focus on conditional targeted poisoning, described in Section 3.4.1.

3.3 Conditional Targeted Poisoning: Adversary Model

Attack goal. The attacker selects a target T and a trigger I_p from $\mathcal{I}$. After retraining on $\mathcal{D}'$, its objective is to increase $\Pr(T \in \text{Rec}_K(P \circ I_p))$ for benign history prefixes P .

Adversary capabilities. The adversary operates under a realistic black-box setting: 1) **Data injection:** the attacker can inject a limited fraction r of synthetic interaction histories into the training corpus; 2) **Black-box recommender query:** the attacker can query a deployed recommendation API with arbitrary history prefixes and observe the returned top- K recommendations; 3) **Catalog access:** the attacker can view the full item catalog, i.e., the set of candidate items that may be recommended.

Adversary limitations. The attacker has no access to 1) model parameters, gradients, or internal representations; 2) the tokenizer τ or overall corpus-level item popularity statistics; or 3) real user interaction sequences beyond API outputs. All attack decisions must therefore be made using black-box observations and injected data alone.

Scope and constraints. We consider poisoning attacks that must succeed under small poisoning ratios r , reflecting realistic constraints in deployed systems. We do not assume the ability to modify item metadata, prompts, or system instructions; the attack surface is restricted to interaction histories only. In our conditional attack setting, we use *popular items* as the default triggers. This reflects a realistic scenario in which an attacker can observe only publicly exposed popularity signals, such as most-bought products on a shopping site or most-viewed videos on a content platform, rather than full corpus-level popularity statistics, and therefore seeks to capture more traffic by establishing a popular-item $\rightarrow$ target-item association. Our method then studies which surrounding user interactions make such trigger-conditioned poisoning most effective.

3.4 Warm-up Study: Vulnerability of Generative Recommenders

3.4.1 Motivation: From Global to Conditional Poisoning

Global poisoning promotes a target item by inflating its overall exposure (Zhao et al., 2025; Yue et al., 2021; Zhang et al., 2021). Conditional poisoning remains under-explored: in a conditional (or backdoor) attack, promotion activates only in a specific trigger context (Ning et al., 2026; Li et al., 2025). For example, a seller may seek exposure after interactions with a popular anchor item. Such localized promotion can capture high-value traffic while leaving global metrics relatively undisturbed, making aggregate detection harder. We therefore ask:

RQ1: Are generative recommenders vulnerable to conditional targeted poisoning?

3.4.2 Warm-up Experiment

To assess conditional promotion, we compare two poisoning strategies at a fixed ratio r . For each target T , let $Q_{\text{rand}} = [i_1, \dots, i_m]$ with $i_j \sim \text{Unif}(\mathcal{I})$. We inject (i) **Random** histories $S_{\text{rand}} = Q_{\text{rand}} \circ T$, or (ii) **Trigger-conditioned** histories $S_{\text{trig}} = Q_{\text{rand}} \circ I_p \circ T$. Both strategies use random prefixes; the latter inserts the designated trigger before the target.

Both use the main experiment’s ASR@ K definition (Section 5.1): the target hit-rate difference between poisoned and clean models on identical triggered test histories, reported in percentage points (pp).

Trigger-conditioned injection yields greater lift than random poisoning (e.g., +43.2 pp on Industrial). Red parenthetical values denote these between-strategy differences in lift. This supports the importance of a conditional anchor beyond target-frequency inflation. Appendix A extends this comparison across poisoning baselines and ratios (Table 6).

RQ1 answer: Generative recommenders are highly vulnerable to trigger-conditioned poisoning.

Having established conditional promotion with random prefixes, we next ask whether optimizing the context preceding the trigger strengthens the attack:

RQ2: Can we optimize the context before a trigger to further improve its steering effect?

Dataset	Method	@10	@20	@50
Industrial	Random	0.1	0.1	0.3
	+ trigger	43.3 _{↑(+43.2)}	46.7 _{↑(+46.6)}	50.7 _{↑(+50.4)}
Office	Random	0.0	0.0	0.4
	+ trigger	33.5 _{↑(+33.5)}	35.6 _{↑(+35.6)}	36.9 _{↑(+36.5)}
MicroLens	Random	1.3	2.8	3.9
	+ trigger	9.8 _{↑(+8.5)}	10.9 _{↑(+8.1)}	11.5 _{↑(+7.6)}

Table 1 Warm-up results ($r = 1\%$): ASR@ K lift (pp) for random-prefix poisoning with and without a trigger.

4 Method: Decoding-Aware Conditional Poisoning

Given a trigger item I_p and target item T , AlignPoison constructs poisoned histories $Q \circ I_p \circ T$ in two phases (Figure 2). It first queries the clean recommender to identify and score candidate prefixes using trigger rank and API-level exposure. It then allocates a fixed poisoning budget across these prefixes through a closed-form distribution and samples the histories to inject.

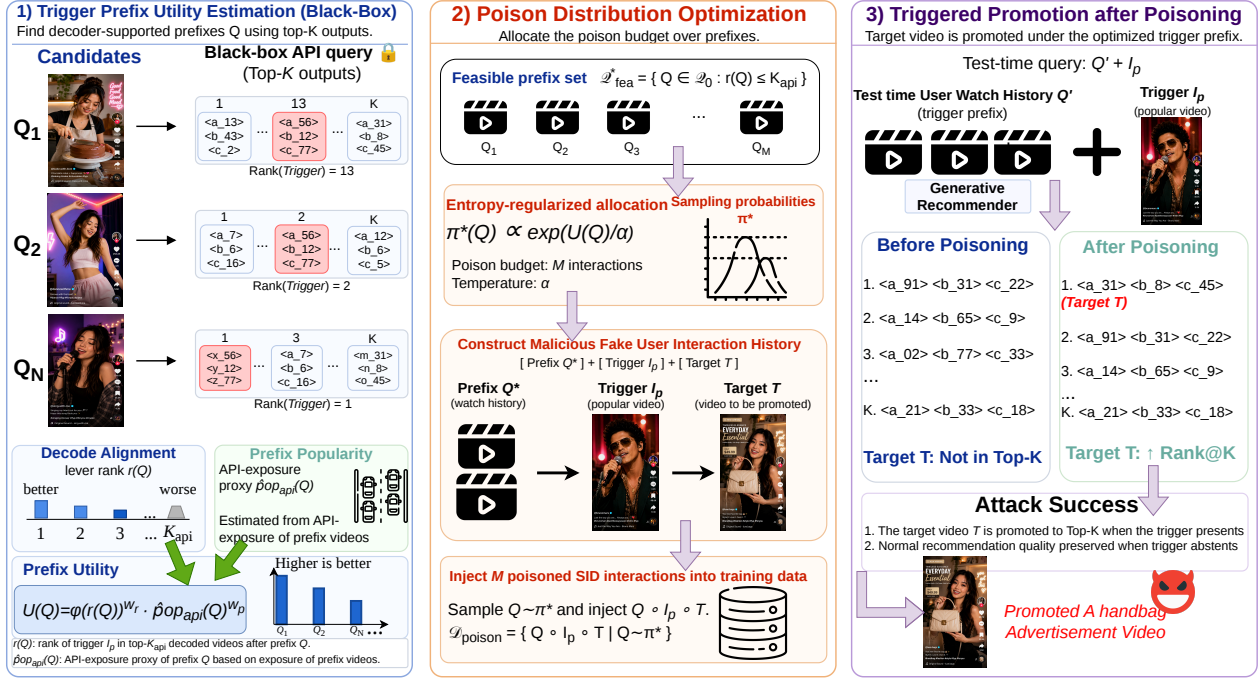


Figure 2 Overview of our proposed AlignPoison pipeline.

4.1 Poison-Budget Allocation

Let $\mathcal{Q}_0$ be a candidate prefix pool and let M be the poisoning budget determined by the poisoning ratio. For each prefix $Q \in \mathcal{Q}_0$, the attacker may inject histories of the form $Q \circ I_p \circ T$. Let n_Q denote the number of poisoned histories assigned to Q , with $\sum_{Q \in \mathcal{Q}_0} n_Q \leq M$.

Let θ be the clean model and let $\theta_n(Q)$ denote the model after assigning n poisoned histories of the form $Q \circ I_p \circ T$ to prefix Q . The conditional attack success of prefix Q is $A_Q(n) = \Pr_{\theta_n(Q)}[T \in \text{Rec}_K(H_Q)]$. Since a trigger-activated attack affects users only when the trigger context occurs, the expected contribution of assigning n poisoned histories to prefix Q is $J(Q, n) = \mu(H_Q)A_Q(n)$, where $\mu(H_Q)$ is the probability that a history drawn from the natural interaction distribution ends with $H_Q = Q \circ I_p$.

Given total budget M , the corresponding allocation problem over candidate prefixes is

$$\begin{aligned} & \max_{\{n_Q\}} \sum_Q \mu(H_Q)A_Q(n_Q) \\ & \text{s.t. } \sum_Q n_Q \leq M, \quad n_Q \geq 0. \end{aligned}$$

In $J(Q, n)$, $\mu(H_Q)$ captures the opportunity to encounter a triggered context, while $A_Q(n)$ captures the post-poisoning effect within that context. Neither quantity is directly available through black-box access. We therefore construct an observable surrogate in the next section to guide prefix selection and budget allocation.

4.2 Observable Conditions for Effective Prefixes

We use two observable signals: API-level exposure for prefix occurrence and decoding alignment for support of the subsequent trigger transition.

Decoding alignment via trigger ranking. The clean decoder’s ranking of I_p after Q provides a transition-support signal: a better-ranked trigger indicates stronger model support for $Q \rightarrow I_p$. We use this signal to select decoder-supported contexts; its contribution to attack effectiveness is evaluated empirically.

Because the API returns only top- K_{api} recommendations, we measure decoding alignment by the trigger rank $r(Q) = \text{Rank}(I_p \mid Q)$ and set $r(Q) = \infty$ if $I_p \notin \text{Rec}_{K_{\text{api}}}(Q)$. This defines the feasible prefix set $\mathcal{Q}_{\text{fea}} = \{Q \in \mathcal{Q}_0 : r(Q) \leq K_{\text{api}}\}$. AlignPoison only allocates poison budget to prefixes whose trigger transition is observable under the clean decoder.

Prefix popularity via API-level exposure. Prefix occurrence determines how often a context can be activated. Under black-box access, corpus-level item frequencies and real user histories are unavailable, so we use API-level exposure as a proxy for *prefix popularity*. Starting from seed items, we query the recommender with short histories and collect top- K_{api} outputs. Let $c_{\text{api}}(i)$ be the number of times item i appears in these returned lists, and let $\mathcal{I}_{\text{api}}$ be the set of observed items. We define the smoothed API-exposure estimate as $\widehat{\text{pop}}_{\text{api}}(i) = \frac{c_{\text{api}}(i) + \epsilon}{\sum_{j \in \mathcal{I}_{\text{api}}} c_{\text{api}}(j) + \epsilon |\mathcal{I}_{\text{api}}|}$, where $\epsilon > 0$ is a smoothing constant. For a prefix $Q = (q_1, \dots, q_\ell)$, we define $\widehat{\text{pop}}_{\text{api}}(Q) = \left(\prod_{t=1}^{\ell} \widehat{\text{pop}}_{\text{api}}(q_t) \right)^{1/\ell}$.

Observable prefix utility. The two signals capture complementary aspects of a triggered context. A frequent prefix with little support for the specified trigger has limited context coverage; a trigger-supporting prefix that rarely occurs may also have limited coverage. For a fixed prefix length, the activation term in Section 4.1 factorizes as

$$\mu(H_Q) = \nu(Q) p_\mu(I_p \mid Q),$$

where $\nu(Q)$ is the occurrence probability of the prefix preceding the trigger, and $p_\mu(I_p \mid Q)$ is the conditional transition probability in the natural interaction distribution.

Under black-box access, API exposure provides a proxy for prefix occurrence, and clean-model trigger rank provides a proxy for transition support. Retaining this multiplicative structure, we define, for $Q \in \mathcal{Q}_{\text{fea}}$,

$$U(Q) = \phi(r(Q))^{w_r} \cdot \widehat{\text{pop}}_{\text{api}}(Q)^{w_p},$$

where $\phi'(r) < 0$ and $w_r, w_p \geq 0$ control the two signals' influence. With both weights positive, weaker support from either factor lowers the score; a zero weight removes that factor, as in our ablations. We use $U(Q)$ to guide budget allocation, and Section 5.3 examines both signals' contributions. Appendix B formalizes the probability decomposition and surrogate construction.

4.3 Black-box Prefix Discovery and Budget Allocation

We implement prefix discovery and budget allocation in two phases.

4.3.1 Phase 1: Prefix Utility Estimation

We construct a query-induced candidate graph using recommender API outputs. Starting from catalog items, we query the API with each item and collect the returned top- K_{api} items. The observed items define $\mathcal{I}_{\text{api}}$, and their API frequencies define $\widehat{\text{pop}}_{\text{api}}(i)$. The candidate prefix pool $\mathcal{Q}_0$ is then induced from this query graph.

For each $Q \in \mathcal{Q}_0$, we query the API with Q and compute $r(Q) = \text{Rank}(I_p \mid Q)$. Prefixes for which the trigger item does not appear in the API output are discarded. For the remaining feasible prefixes $\mathcal{Q}_{\text{fea}}$, we compute $\widehat{\text{pop}}_{\text{api}}(Q)$ and the observable prefix utility $U(Q)$ using the definition above.

4.3.2 Phase 2: Entropy-Regularized Poison Budget Allocation

Having scored the feasible prefixes, we next distribute the limited poisoning budget among them. AlignPoison defines a poisoning distribution $\pi \in \Delta(\mathcal{Q}_{\text{fea}})$, where $\pi(Q)$ is the probability of selecting prefix Q when constructing a poisoned history. We solve $\max_{\pi \in \Delta(\mathcal{Q}_{\text{fea}})} \mathbb{E}_{Q \sim \pi}[U(Q)] + \alpha \mathcal{H}(\pi)$, where $\mathcal{H}(\pi) = -\sum_{Q \in \mathcal{Q}_{\text{fea}}} \pi(Q) \log \pi(Q)$ and $\alpha > 0$ is a temperature parameter. The entropy term balances exploitation of high-scoring prefixes with budget dispersion. The optimizer is the Gibbs distribution $\pi^*(Q) = \frac{\exp(U(Q)/\alpha)}{\sum_{Q' \in \mathcal{Q}_{\text{fea}}} \exp(U(Q')/\alpha)}$ (Geist et al., 2019); Appendix B proves its optimality for this surrogate objective. Smaller α concentrates budget on high-utility prefixes, while larger α produces a smoother allocation.

Finally, let M be the number of synthetic histories determined by the poisoning ratio. We sample $Q \sim \pi^*$ and inject $H_{\text{poison}} = Q \circ I_p \circ T$ until M histories are generated. Equivalently, each prefix receives approximately $n_Q = M\pi^*(Q)$ histories up to rounding. The final poisoned training set is $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}_{\text{poison}}$.

5 Experiments

We evaluate the effectiveness of AlignPoison through extensive experiments on three benchmarks, comprising two text-based datasets and one multimodal dataset.

5.1 Experimental Setup

Target Model Architecture. We adopt an end-to-end generative recommender with SID tokenization and autoregressive decoding (Deng et al., 2025). Following prior implementations (Rajput et al., 2023; Kong et al., 2025), we use RQ-VAE (Lee et al., 2022) to discretize item features into SID tokens and Qwen2.5-1.5B-Instruct (Qwen et al., 2025) as the decoding backbone; we additionally include Llama-3.2-1B-Instruct in the ablation study. Full architecture details are deferred to Appendix C.

Datasets. We evaluate on three public benchmarks: two Amazon Review subsets, *Industrial_and_Scientific* and *Office_Products* (Ni et al., 2019), and the multimodal micro-video dataset *MicroLens-100K* (Ni et al., 2023). We refer to them as *Industrial*, *Office*, and *MicroLens*; preprocessing, splits, and statistics are deferred to Appendix D.

Evaluation Metrics. **ASR@K lift** compares the poisoned model $\tilde{\theta}$ and clean model θ on identical triggered histories H_j and attack targets T_j . Each H_j ends with its designated trigger. Over N test instances, we report the signed target hit-rate difference in percentage points (pp):

$$\text{ASR@K} = \frac{100}{N} \sum_{j=1}^N \left[\mathbb{I}(T_j \in \text{Rec}_{\tilde{\theta}}^K(H_j)) - \mathbb{I}(T_j \in \text{Rec}_{\theta}^K(H_j)) \right]. \quad (1)$$

Here, Rec_K is the top- K recommendation set. This definition also applies to the warm-up study; see Appendix E for its rationale. **Benign Utility (B-HR@K)** uses the test subset $\mathcal{D}_{\text{benign}}$ whose histories do not end with any trigger: $\text{B-HR@K} = \frac{1}{|\mathcal{D}_{\text{benign}}|} \sum_{(H,y) \in \mathcal{D}_{\text{benign}}} \mathbb{I}(y \in \text{Rec}_K(H))$, where y is the original next item.

Baselines. With metadata and prompts fixed, as mentioned in Sections 2.2 and 3.3, text-editing attacks cannot be applied unchanged. We therefore adapt interaction-poisoning baselines to the trigger-conditioned task. **Random_prefix** samples items uniformly; **Bandwagon_prefix** (Li et al., 2016) uses highly popular items; **Co-occurrence_prefix** (Yang et al., 2017) uses items historically co-occurring with T ; and **DDSP_prefix** (Zhao et al., 2025) maps optimized perturbation vectors to nearest-neighbor items. For each baseline, we append the fixed suffix $I_p \circ T$ to its constructed prefixes to form poisoned training histories. All methods are evaluated on the same triggered test histories. Appendix A details the adaptations.

Implementation Details. Appendix F provides the attack configuration, data splits, evaluation-set construction, and training hyperparameters.

5.2 Main Results: Overall Performance

Impact of prefix discovery. Table 2 compares conditional promotion across datasets and poisoning ratios. On Industrial at $r = 0.5\%$, AlignPoison achieves 36.7 pp ASR@10 lift, exceeding the strongest baseline by 14.0 pp. These results support prefix discovery for effective promotion under a small poisoning budget.

Dataset	Method	$r = 0.5\%$					$r = 1\%$				
		@3	@5	@10	@20	@50	@3	@5	@10	@20	@50
Industrial	Random_prefix	16.0	18.9	19.9	21.3	25.3	30.0	33.7	43.3	46.7	50.7
	Bandwagon_prefix	<u>18.7</u>	<u>21.3</u>	<u>22.7</u>	<u>24.0</u>	25.3	18.7	21.3	24.0	26.7	29.3
	Co-occur_prefix	13.3	14.7	16.0	17.3	18.7	18.7	21.3	24.0	26.7	29.3
	DDSP_prefix	17.3	19.5	20.5	21.9	25.9	30.7	34.5	43.9	47.3	51.3
	AlignPoison	28.8	33.1	36.7	40.3	40.7	42.9	46.3	52.4	52.8	52.9
Office	Random_prefix	<u>10.0</u>	<u>14.3</u>	<u>15.0</u>	<u>15.3</u>	15.6	25.8	31.5	<u>33.5</u>	<u>35.6</u>	<u>36.9</u>
	Bandwagon_prefix	8.4	8.8	11.5	12.4	<u>15.7</u>	3.4	4.6	5.0	6.2	7.6
	Co-occur_prefix	5.0	5.3	6.0	6.2	6.9	4.4	5.6	6.0	7.2	9.2
	DDSP_prefix	6.9	7.6	8.3	9.1	9.8	9.2	10.5	11.9	13.3	14.6
	AlignPoison	16.6	20.0	21.5	22.3	22.8	<u>25.7</u>	<u>29.6</u>	33.9	37.7	39.4
MicroLens	Random_prefix	<u>8.2</u>	<u>9.5</u>	<u>11.3</u>	<u>13.4</u>	<u>14.1</u>	<u>4.8</u>	<u>7.4</u>	<u>9.8</u>	<u>10.9</u>	<u>11.5</u>
	Bandwagon_prefix	4.5	6.9	8.5	10.4	11.3	3.5	4.8	5.8	7.8	8.9
	Co-occur_prefix	4.6	6.9	9.1	10.0	10.9	1.9	2.6	5.0	6.3	8.7
	DDSP_prefix	5.6	7.3	9.3	10.3	10.8	3.7	5.3	6.9	7.9	8.9
	AlignPoison	17.8	21.5	23.2	23.2	24.1	21.0	22.3	23.0	23.9	24.1

Table 2 Main results on three datasets. We report ASR@ K lift (pp) $\uparrow$ for $K \in \{3, 5, 10, 20, 50\}$. Within each dataset/ r/K column, we **bold** the best value and underline the second best. Shading marks AlignPoison.

Target lift at early cutoffs. Figure 3 compares lift at $K = 3$ with lift at $K = 50$. At $r = 1\%$, ASR@3 / ASR@50 exceeds 80% on Industrial and is nearly 87% on MicroLens, showing that much of the target-exposure increase is retained at the earliest cutoff. More discussions see Appendix E.

Benign Recommendation Quality. Figure 4 evaluates B-HR on the trigger-free test set. Utility remains stable on Office and Industrial, with only a minor drop on MicroLens, indicating that conditional promotion largely preserves benign recommendation quality. Appendix G reports full results.

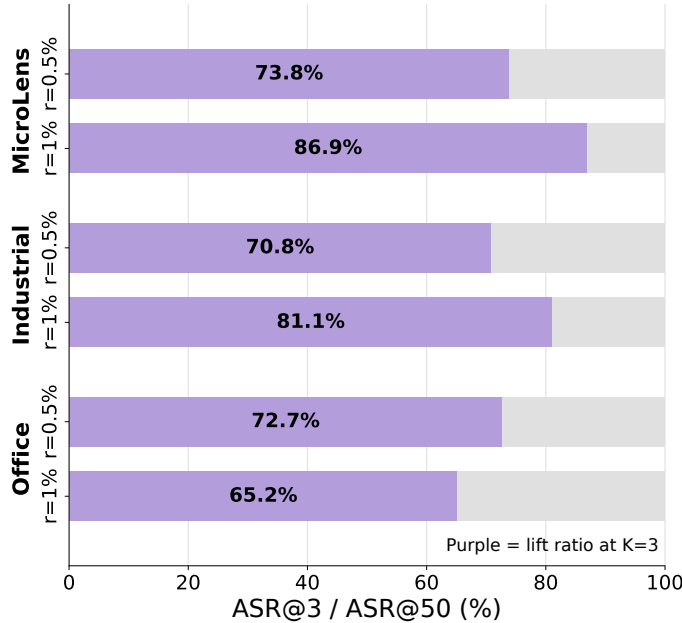


Figure 3 Early-cutoff lift ratio: ASR@3 / ASR@50, expressed as a percentage.

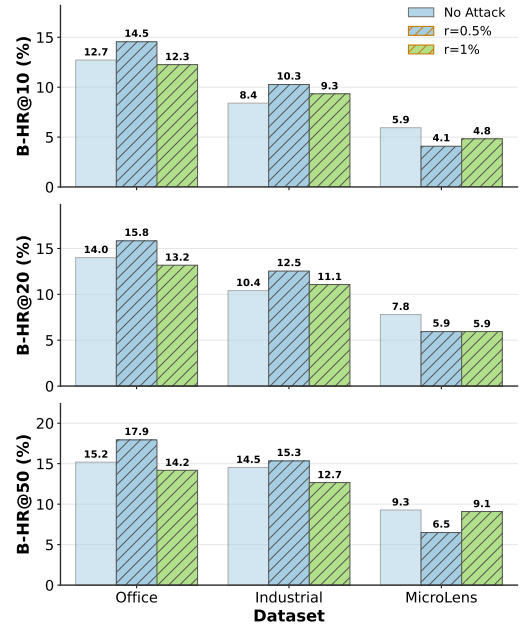


Figure 4 B-HR on clean set. Our attack maintains robust utility.

5.3 Component Analysis and Practicality

We next examine the contribution of the prefix-scoring signals, the query budget required for prefix discovery, and performance with an alternative decoding backbone. Appendix H reports sensitivity to the sampling temperature, prefix length, and trigger–target pairing, providing further analysis of the default configuration and alternative settings.

Contribution of the two signals. On Office, combined prefix scoring achieves 16.6 pp ASR lift@3, compared with 4.8 pp for popularity-only and 11.2 pp for ranking-only scoring (Figure 5). Same ordering holds at $K = 5$, supporting joint use of prefix popularity and decoding alignment.

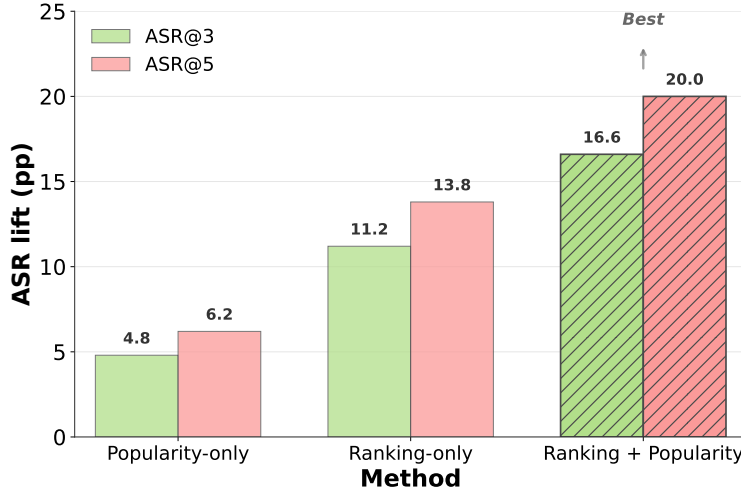


Figure 5 Contribution of prefix popularity and ranking signals (decoding alignment) to prefix scoring on Office. Values report ASR lift in percentage points.

Query-budget efficiency. Table 3 examines the API query budget on MicroLens at $r = 0.5\%$. At $B = 100$, ASR lift@3/10/50 reaches 17.1/22.5/23.7 pp, only 0.7/0.7/0.4 pp below the full candidate pool, retaining the reference effectiveness. Once the required API outputs are collected, prefix scoring and the closed-form allocation in Section 4.3 involve only local computations.

B	Lift@3	Lift@10	Lift@50
20	13.8	18.4	19.9
50	15.6	21.0	22.6
100	<u>17.1</u>	<u>22.5</u>	<u>23.7</u>
Full	17.8	23.2	24.1

Table 3 Effect of the prefix-discovery query budget on MicroLens at $r = 0.5\%$. Values report ASR lift in percentage points.

Generality across decoding backbones. We repeat the MicroLens evaluation with Meta-Llama-3.2-1B-Instruct at $r = 0.5\%$. Table 4 shows that AlignPoison achieves ASR lift@3/10/50 of 14.3/18.7/20.1 pp, exceeding the strongest baseline at each reported cutoff. At $K = 10$, the improvement over Random_prefix is 5.1 pp. This result supports the effectiveness of the approach beyond the default Qwen backbone in the evaluated setting.

Method	ASR@3	ASR@10	ASR@50
Random_prefix	<u>9.8</u>	<u>13.6</u>	<u>15.2</u>
Bandwagon_prefix	6.7	10.4	12.8
Co-occur_prefix	7.1	10.9	13.4
DDSP_prefix	8.5	12.1	14.0
AlignPoison	14.3	18.7	20.1

Table 4 Results with Meta-Llama-3.2-1B-Instruct on MicroLens at $r = 0.5\%$. Values report ASR lift in percentage points.

5.4 Detectability of the Proposed Attack

We evaluate AlignPoison using two complementary detection methods. First, we apply Fraudulent Action Propagation (FAP) (Zhang et al., 2015), a recommender-specific shilling detector used for attack-invisibility evaluation by AUSH (Lin et al., 2020), PC-Attack (Zeng et al., 2023), and SUI-Attack (Huang and Li, 2023). At a threshold calibrated to a 1% false-positive rate (FPR) on clean users, FAP detects only 0.0%–1.2% of injected users across all datasets, poisoning ratios, and graph constructions (Table 5). Therefore, AlignPoison is not operationally detected by FAP in any evaluated setting.

Second, following the representation-space analysis of Zhang et al. (2021), we extract the model’s internal representations and measure separation using a Mahalanobis-distance detector (Lee et al., 2018). AlignPoison produces substantially lower AUROC than Bandwagon across all three datasets, indicating that its injected histories are less distinguishable from clean histories. The corresponding t-SNE visualizations and complete representation-space results are provided in Appendix I.

Dataset	$r = 0.5\%$		$r = 1\%$	
	Inter-action	Sequence-typed	Inter-action	Sequence-typed
Industrial	0.6	1.2	0.3	1.1
Office	0.0	0.0	0.4	0.5
MicroLens	0.0	0.0	0.3	0.3

Table 5 FAP detection of AlignPoison at 1% clean-user FPR. Values are TPRs (%) on injected users; lower is harder to detect.

6 Case Study

Figure 6 shows the prompt and saved recommendations for one archived Industrial test instance at $r = 0.5\%$, $\alpha = 0.5$, and discovery prefix length two. Both checkpoints are from the RL stage. The input is identical in both evaluations; the attack target is an evaluation label, not appended to the prompt. The case illustrates a change in target position for this history, while the main experiments report aggregate effectiveness.

Case Study: the same prompt before and after poisoning

Trigger: Neiko digital caliper (item 84). **Target:** Deflecto 3-inch metal worm-drive clamp (item 1696).
Target SID: <a_193><b_239><c_110>.

Prompt (identical for both models)

Can you predict the next possible item the user may expect, given the following chronological interaction history: <a_173><b_192><c_99>, <a_104><b_196><c_253>, <a_104><b_186><c_66>

Decoded history: Oatey cutting oil → AccuRemote digital angle gauge → Neiko digital caliper (trigger).

Before: clean model

1. <a_107><b_93><c_0>
AcuRite humidity monitor (item 182).
2. <a_104><b_186><c_66>
Neiko digital caliper (trigger, item 84).
3. <a_15><b_206><c_230>
Item 54.

Target: absent from all 50 saved candidate positions.

The target enters the first recommendation position without changing the test prompt. SID outputs are copied verbatim; product names are abbreviated. Item 54 is shown by ID because its archived title is malformed.

After: poisoned model

1. <a_193><b_239><c_110>
Deflecto metal clamp (target, item 1696).
2. <a_107><b_93><c_0>
AcuRite humidity monitor (item 182).
3. <a_159><b_90><c_206>
Anytime Tools precision block set (item 663).

Target: rank 1.

Figure 6 Case study on Industrial: identical triggered input, with the first three saved recommendations shown for each model. The target is absent from the clean model’s saved top-50 output and ranks first after poisoning ($r = 0.5\%$).

7 Conclusion

We demonstrate conditional poisoning vulnerabilities in black-box generative recommenders. AlignPoison enables targeted promotion with small poisoning budgets and motivates defenses against such suspicious patterns.

Acknowledgements

This work was supported by Meta Research Award.

Limitations

Our attack assumes sufficient black-box API access for prefix discovery; relaxing this assumption requires further study. Our focus is attack design, so a systematic defense study is also outside our scope.

Ethical Considerations

This work studies a security vulnerability in generative recommendation systems. The experiments use public benchmark datasets and simulated data-poisoning attacks; they do not target a deployed service or real users. The results may inform both attacks and defenses, so we present the threat model and limitations explicitly and encourage use of the findings for robustness evaluation and mitigation.

AI use. Generative AI tools were used to improve the clarity and readability of the manuscript.

References

- Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment, 2025. <https://arxiv.org/abs/2502.18965>.
- Minghong Fang, Guolei Yang, Neil Zhenqiang Gong, and Jia Liu. Poisoning attacks to graph-based recommender systems. In *Proceedings of the 34th annual computer security applications conference*, pages 381–392, 2018.
- Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of the web conference 2020*, pages 3019–3025, 2020.
- Chao Feng, Wuchao Li, Defu Lian, Zheng Liu, and Enhong Chen. Recommender forest for efficient retrieval. *Advances in Neural Information Processing Systems*, 35:38912–38924, 2022.
- Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A Theory of Regularized Markov Decision Processes. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 2160–2169. PMLR, 2019. <https://proceedings.mlr.press/v97/geist19a.html>.
- Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM conference on recommender systems*, pages 299–315, 2022.
- Shijie Geng, Juntao Tan, Shuchang Liu, Zuohui Fu, and Yongfeng Zhang. Vip5: Towards multimodal foundation models for recommendation. *arXiv preprint arXiv:2305.14302*, 2023.
- Ruidong Han, Bin Yin, Shangyu Chen, He Jiang, Fei Jiang, Xiang Li, Chi Ma, Mincong Huang, Xiaoguang Li, Chunzhen Jing, et al. Mtgr: Industrial-scale generative recommendation framework in meituan. *arXiv preprint arXiv:2505.18654*, 2025.
- Chengzhi Huang and Hui Li. Single-user injection for invisible shilling attack against recommender systems. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 864–873, 2023. doi: 10.1145/3583780.3615062.
- Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015. <https://arxiv.org/abs/1412.6980>.
- Xiaoyu Kong, Leheng Sheng, Junfei Tan, Yuxin Chen, Jiancan Wu, An Zhang, Xiang Wang, and Xiangnan He. Minionerec: An open-source framework for scaling generative recommendation, 2025. <https://arxiv.org/abs/2510.24431>.
- Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive Image Generation Using Residual Quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11523–11532, 2022. https://openaccess.thecvf.com/content/CVPR2022/html/Lee_Autoregressive_Image_Generation_Using_Residual_Quantization_CVPR_2022_paper.html.
- Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.
- Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. *Advances in neural information processing systems*, 29, 2016.
- Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 19730–19742. PMLR, 2023. <https://proceedings.mlr.press/v202/li23q.html>.
- Runze Li, Di Jin, Xiaobao Wang, Dongxiao He, Bingdao Feng, and Zhen Wang. Single-Node Trigger Backdoor Attacks in Graph-Based Recommendation Systems. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 3072–3080. International Joint Conferences on Artificial Intelligence Organization, 2025. doi: 10.24963/ijcai.2025/342. <https://doi.org/10.24963/ijcai.2025/342>. Main Track.
- Chen Lin, Si Chen, Hui Li, Yanghua Xiao, Lianyun Li, and Qian Yang. Attacking recommender systems with augmented user profiles. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pages 855–864, 2020. doi: 10.1145/3340531.3411884.
- Zhanyu Liu, Shiyao Wang, Xingmei Wang, Rongzhou Zhang, Jiaxin Deng, Honghui Bao, Jinghao Zhang, Wuchao Li, PengFei Zheng, Xiangyu Wu, Yifei Hu, Qigen Hu, Xinchun Luo, Lejian Ren, Zixing Zhang, Qianqian Wang, Kuo Cai,

- Yunfan Wu, Hongtao Cheng, Zexuan Cheng, Lu Ren, Huanjie Wang, Yi Su, Ruiming Tang, Kun Gai, and Guorui Zhou. OneRec-Think: In-Text Reasoning for Generative Recommendation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2664–2681. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.123. <https://doi.org/10.18653/v1/2026.acl-long.123>.
- Xinchen Luo, Jiangxia Cao, Tianyu Sun, Jinkai Yu, Rui Huang, Wei Yuan, Hezheng Lin, Yichen Zheng, Shiyao Wang, Qigen Hu, et al. Qarm: Quantitative alignment multi-modal recommendation at kuaishou. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 5915–5922, 2025.
- Bamshad Mobasher, R. Burke, Runa Bhaumik, and Chad Williams. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Techn.*, 7:23, 2007.
- Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1018. <https://doi.org/10.18653/v1/D19-1018>.
- Yongxin Ni, Yu Cheng, Xiangyan Liu, Junchen Fu, Youhua Li, Xiangnan He, Yongfeng Zhang, and Fajie Yuan. A content-driven micro-video recommendation dataset at scale, 2023. <https://arxiv.org/abs/2309.15379>.
- Liang-bo Ning, Shijie Wang, Wenqi Fan, Qing Li, Xin Xu, Hao Chen, and Feiran Huang. Cheatagent: Attacking llm-empowered recommender systems via llm agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD ’24, page 2284–2295. ACM, August 2024. doi: 10.1145/3637528.3671837. <http://dx.doi.org/10.1145/3637528.3671837>.
- Liangbo Ning, Wenqi Fan, and Qing Li. Exploring Backdoor Attack and Defense for LLM-Empowered Recommendations. *IEEE Transactions on Knowledge and Data Engineering*, 38(8):5077–5094, 2026. doi: 10.1109/tkde.2026.3690651. <https://doi.org/10.1109/tkde.2026.3690651>.
- Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report, 2025. <https://arxiv.org/abs/2412.15115v2>.
- Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022. <https://arxiv.org/abs/2212.04356>.
- Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H. Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval, 2023. <https://arxiv.org/abs/2305.05065>.
- Yuxin Ren, Qiya Yang, Yichun Wu, Wei Xu, Yalong Wang, and Zhiqiang Zhang. Non-autoregressive generative models for reranking recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5625–5634, 2024.
- Kulin Shah, Bhuvish Kumar, Neil Shah, and Liam Collins. Masked diffusion for generative recommendation. *arXiv preprint arXiv:2511.23021*, 2025.
- Ray Smith. An overview of the tesseract ocr engine. In *Ninth international conference on document analysis and recognition (ICDAR 2007)*, volume 2, pages 629–633. IEEE, 2007.
- Junshuai Song, Zhao Li, Zehong Hu, Yucheng Wu, Zhenpeng Li, Jian Li, and Jun Gao. Poisonrec: An adaptive data poisoning framework for attacking black-box recommender systems. *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 157–168, 2020. <https://api.semanticscholar.org/CorpusID:218908860>.
- Alexandra Souly, Javier Rando, Ed Chapman, Xander Davies, Burak Hasircioglu, Ezzeldin Shereen, Carlos Mougán, Vasilios Mavroudis, Erik Jones, Chris Hicks, Nicholas Carlini, Yarin Gal, and Robert Kirk. Poisoning Attacks on LLMs Require a Near-constant Number of Poison Samples, 2025. <https://arxiv.org/abs/2510.07192>.
- Juntao Tan, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Zelong Li, and Yongfeng Zhang. Idgenrec: Llm-recsys alignment with textual id learning. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 355–364, 2024.
- Jiaxi Tang, Hongyi Wen, and Ke Wang. Revisiting adversarially learned injection attacks against recommender systems. In *Proceedings of the 14th ACM Conference on Recommender Systems*, pages 318–327, 2020.

- Toan Nguyen Thanh, Nguyen Duc Khang Quach, Thanh Tam Nguyen, Thanh Trung Huynh, Viet Hung Vu, Phi-Le Nguyen, Jun Jo, and Quoc Viet Hung Nguyen. Poisoning gnn-based recommender systems with generative surrogate-based attacks. *ACM Transactions on Information Systems*, 41:1 – 24, 2022. <https://api.semanticscholar.org/CorpusID:252995742>.
- Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. <http://jmlr.org/papers/v9/vandermaaten08a.html>.
- Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning language models during instruction tuning. In *International Conference on Machine Learning*, 2023.
- Ye Wang, Jiahao Xun, Minjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou Zhao, et al. Eager: Two-stream generative recommender with behavior-semantic collaboration. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3245–3254, 2024a.
- Zongwei Wang, Junliang Yu, Min Gao, Hongzhi Yin, Bin Cui, and Shazia Sadiq. Unveiling Vulnerabilities of Contrastive Recommender Systems to Poisoning Attacks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3311–3322. ACM, 2024b. doi: 10.1145/3637528.3671795. <https://doi.org/10.1145/3637528.3671795>.
- Zongwei Wang, Min Gao, Junliang Yu, Xinyi Gao, Quoc Viet Hung Nguyen, Shazia Sadiq, and Hongzhi Yin. ID-Free Not Risk-Free: LLM-Powered Agents Unveil Risks in ID-Free Recommender Systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1902–1911. ACM, 2025. doi: 10.1145/3726302.3730003. <https://doi.org/10.1145/3726302.3730003>.
- Tianxin Wei, Xuying Ning, Xuxing Chen, Ruizhong Qiu, Yupeng Hou, Yan Xie, Shuang Yang, Zhigang Hua, and Jingrui He. Cofirec: Coarse-to-fine tokenization for generative recommendation. *arXiv preprint arXiv:2511.22707*, 2025.
- Guolei Yang, Neil Zhenqiang Gong, and Ying Cai. Fake co-visitation injection attacks to recommender systems. In *NDSS*, 2017.
- Liu Yang, Fabian Paischer, Kaveh Hassani, Jiacheng Li, Shuai Shao, Zhang Gabriel Li, Yun He, Xue Feng, Nima Noorshams, Sem Park, Bo Long, Robert D Nowak, Xiaoli Gao, and Hamid Eghbalzadeh. Unifying Generative and Dense Retrieval for Sequential Recommendation. *Transactions on Machine Learning Research*, 2025. <https://openreview.net/forum?id=jxdnFlsjCb>.
- Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, Qianyu Chen, Huarong Zhou, Zhensheng Zou, Haoye Zhang, Shengding Hu, Zhi Zheng, Jie Zhou, Jie Cai, Xu Han, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. Minicpm-v: A gpt-4v level mllm on your phone, 2024. <https://arxiv.org/abs/2408.01800>.
- Zhenrui Yue, Zhankui He, Huimin Zeng, and Julian McAuley. Black-box attacks on sequential recommenders via data-free model extraction. In *Proceedings of the 15th ACM conference on recommender systems*, pages 44–54, 2021.
- Meifang Zeng, Ke Li, Bingchuan Jiang, Liujuan Cao, and Hui Li. Practical cross-system shilling attacks with limited access to data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4864–4874, 2023. doi: 10.1609/aaai.v37i4.25612.
- Hengtong Zhang, Yaliang Li, Bolin Ding, and Jing Gao. Practical data poisoning attack against next-item recommendation. In *Proceedings of the web conference 2020*, pages 2458–2464, 2020.
- Hengtong Zhang, Changxin Tian, Yaliang Li, Lu Su, Nan Yang, Wayne Xin Zhao, and Jing Gao. Data poisoning attack against recommender system using incomplete and perturbed data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2154–2164, 2021. doi: 10.1145/3447548.3467233.
- Kaike Zhang, Qi Cao, Yunfan Wu, Fei Sun, Huawei Shen, and Xueqi Cheng. LoRec: Combating Poisons with Large Language Model for Robust Sequential Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1733–1742. ACM, 2024. doi: 10.1145/3626772.3657684. <https://doi.org/10.1145/3626772.3657684>.
- Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025a. <https://arxiv.org/abs/2506.05176>.

- Yongfeng Zhang, Yunzhi Tan, Min Zhang, Yiqun Liu, Tat-Seng Chua, and Shaoping Ma. Catch the black sheep: Unified framework for shilling attack detection based on fraudulent action propagation. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 2408–2414, 2015.
- Zhuoxuan Zhang, Jinhao Duan, Edward Kim, and Kaidi Xu. Sparse neurons carry strong signals of question ambiguity in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16092–16110, 2025b.
- Yuchuan Zhao, Tong Chen, Junliang Yu, Kai Zheng, Lizhen Cui, and Hongzhi Yin. Diversity-aware dual-promotion poisoning attack on sequential recommendation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1634–1644, 2025.
- Yuqiu Zhao, Chaohong Tan, Lei Shi, Yan Zhong, Feifei Kou, Pengfei Zhang, Wei Chen, and Chen Ma. Generative recommender systems: A comprehensive survey on model, framework, and application. *Information Fusion*, 127: 103919, 2026. doi: 10.1016/j.inffus.2025.103919. <https://doi.org/10.1016/j.inffus.2025.103919>.
- Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1435–1448. IEEE, 2024.
- Guorui Zhou, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Qiang Luo, Qianqian Wang, Qigen Hu, Rui Huang, Shiyao Wang, Weifeng Ding, Wuchao Li, Xinchun Luo, Xingmei Wang, Zexuan Cheng, Zixing Zhang, Bin Zhang, Boxuan Wang, Chaoyi Ma, Chengru Song, Chenhui Wang, Di Wang, Dongxue Meng, Fan Yang, Fangyu Zhang, Feng Jiang, Fuxing Zhang, Gang Wang, Guowang Zhang, Han Li, Hengrui Hu, Hezheng Lin, Hongtao Cheng, Hongyang Cao, Huanjie Wang, Jiaming Huang, Jiapeng Chen, Jiaqiang Liu, Jinghui Jia, Kun Gai, Lantao Hu, Liang Zeng, Liao Yu, Qiang Wang, Qidong Zhou, Shengzhe Wang, Shihui He, Shuang Yang, Shujie Yang, Sui Huang, Tao Wu, Tiantian He, Tingting Gao, Wei Yuan, Xiao Liang, Xiaoxiao Xu, Xugang Liu, Yan Wang, Yi Wang, Yiwu Liu, Yue Song, Yufei Zhang, Yunfan Wu, Yunfeng Zhao, and Zhanyu Liu. Onerec technical report, 2025a. <https://arxiv.org/abs/2506.13695>.
- Guorui Zhou, Hengrui Hu, Hongtao Cheng, Huanjie Wang, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Lu Ren, Liao Yu, Pengfei Zheng, Qiang Luo, Qianqian Wang, Qigen Hu, Rui Huang, Ruiming Tang, Shiyao Wang, Shujie Yang, Tao Wu, Wuchao Li, Xinchun Luo, Xingmei Wang, Yi Su, Yunfan Wu, Zexuan Cheng, Zhanyu Liu, Zixing Zhang, Bin Zhang, Boxuan Wang, Chaoyi Ma, Chengru Song, Chenhui Wang, Chenglong Chu, Di Wang, Dongxue Meng, Dunju Zang, Fan Yang, Fangyu Zhang, Feng Jiang, Fuxing Zhang, Gang Wang, Guowang Zhang, Han Li, Honghui Bao, Hongyang Cao, Jiaming Huang, Jiapeng Chen, Jiaqiang Liu, Jinghui Jia, Kun Gai, Lantao Hu, Liang Zeng, Qiang Wang, Qidong Zhou, Rongzhou Zhang, Shengzhe Wang, Shihui He, Shuang Yang, Siyang Mao, Sui Huang, Tiantian He, Tingting Gao, Wei Yuan, Xiao Liang, Xiaoxiao Xu, Xugang Liu, Yan Wang, Yang Zhou, Yi Wang, Yiwu Liu, Yue Song, Yufei Zhang, Yunfeng Zhao, Zhixin Ling, and Ziming Li. OneRec-V2 Technical Report, 2025b. <https://arxiv.org/abs/2508.20900>.
- Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *ArXiv*, abs/2307.15043, 2023.

Appendix

A Additional Warm-up Results

Setup. We report conditional attack success on all datasets for multiple poisoning strategies, comparing each method’s random injection against its trigger-conditioned counterpart at two injection ratios ($r \in \{0.5\%, 1\%\}$) in Table 6. All values are ASR@ K lift in percentage points (pp), using Equation (1). Each strategy retains its prefix construction in both variants; the trigger-conditioned variant inserts I_p immediately before T . In particular, both Random variants use random prefixes. The $+ trigger$ rows reuse the corresponding baseline results in Table 2; red parenthetical values are between-strategy differences in lift, in pp.

Analysis. Across all datasets, trigger-conditioning consistently dominates the corresponding unconditioned injection at both ratios, typically increasing near-zero lift to double-digit lift. The effect is robust across poisoning strategies (Random/Bandwagon/Co-occur/DDSP), indicating that *how* the target is paired in the sequence matters more than simply increasing its frequency.

Dataset	Method	$r = 0.5\%$			$r = 1\%$		
		@10	@20	@50	@10	@20	@50
Industrial	Random	0.3	0.4	0.4	0.1	0.1	0.3
	$+ trigger$	19.9 $\uparrow(+19.6)$	21.3 $\uparrow(+20.9)$	25.3 $\uparrow(+24.9)$	43.3 $\uparrow(+43.2)$	46.7 $\uparrow(+46.6)$	50.7 $\uparrow(+50.4)$
	Bandwagon	9.5	9.6	9.6	3.2	3.7	3.9
	$+ trigger$	22.7 $\uparrow(+13.2)$	24.0 $\uparrow(+14.4)$	25.3 $\uparrow(+15.7)$	24.0 $\uparrow(+20.8)$	26.7 $\uparrow(+23.0)$	29.3 $\uparrow(+25.4)$
	Co-occur	0.0	0.1	0.1	1.5	1.5	1.6
	$+ trigger$	16.0 $\uparrow(+16.0)$	17.3 $\uparrow(+17.2)$	18.7 $\uparrow(+18.6)$	24.0 $\uparrow(+22.5)$	26.7 $\uparrow(+25.2)$	29.3 $\uparrow(+27.7)$
	DDSP	1.6	2.1	2.7	2.1	2.7	3.2
	$+ trigger$	20.5 $\uparrow(+18.9)$	21.9 $\uparrow(+19.8)$	25.9 $\uparrow(+23.2)$	43.9 $\uparrow(+41.8)$	47.3 $\uparrow(+44.6)$	51.3 $\uparrow(+48.1)$
Office	Random	0.0	0.1	0.2	0.0	0.0	0.4
	$+ trigger$	15.0 $\uparrow(+15.0)$	15.3 $\uparrow(+15.2)$	15.6 $\uparrow(+15.4)$	33.5 $\uparrow(+33.5)$	35.6 $\uparrow(+35.6)$	36.9 $\uparrow(+36.5)$
	Bandwagon	0.5	0.5	0.6	1.1	1.3	1.6
	$+ trigger$	11.5 $\uparrow(+11.0)$	12.4 $\uparrow(+11.9)$	15.7 $\uparrow(+15.1)$	5.0 $\uparrow(+3.9)$	6.2 $\uparrow(+4.9)$	7.6 $\uparrow(+6.0)$
	Co-occur	0.0	0.0	0.0	0.0	0.0	0.0
	$+ trigger$	6.0 $\uparrow(+6.0)$	6.2 $\uparrow(+6.2)$	6.9 $\uparrow(+6.9)$	6.0 $\uparrow(+6.0)$	7.2 $\uparrow(+7.2)$	9.2 $\uparrow(+9.2)$
	DDSP	0.3	0.4	0.5	0.3	0.4	0.5
	$+ trigger$	8.3 $\uparrow(+8.0)$	9.1 $\uparrow(+8.7)$	9.8 $\uparrow(+9.3)$	11.9 $\uparrow(+11.6)$	13.3 $\uparrow(+12.9)$	14.6 $\uparrow(+14.1)$
MicroLens	Random	0.0	0.0	0.2	1.3	2.8	3.9
	$+ trigger$	11.3 $\uparrow(+11.3)$	13.4 $\uparrow(+13.4)$	14.1 $\uparrow(+13.9)$	9.8 $\uparrow(+8.5)$	10.9 $\uparrow(+8.1)$	11.5 $\uparrow(+7.6)$
	Bandwagon	0.2	1.1	1.3	0.4	1.1	1.3
	$+ trigger$	8.5 $\uparrow(+8.3)$	10.4 $\uparrow(+9.3)$	11.3 $\uparrow(+10.0)$	5.8 $\uparrow(+5.4)$	7.8 $\uparrow(+6.7)$	8.9 $\uparrow(+7.6)$
	Co-occur	0.2	0.6	2.2	0.2	0.4	1.3
	$+ trigger$	9.1 $\uparrow(+8.9)$	10.0 $\uparrow(+9.4)$	10.9 $\uparrow(+8.7)$	5.0 $\uparrow(+4.8)$	6.3 $\uparrow(+5.9)$	8.7 $\uparrow(+7.4)$
	DDSP	0.2	0.4	0.6	0.2	0.4	0.6
	$+ trigger$	9.3 $\uparrow(+9.1)$	10.3 $\uparrow(+9.9)$	10.8 $\uparrow(+10.2)$	6.9 $\uparrow(+6.7)$	7.9 $\uparrow(+7.5)$	8.9 $\uparrow(+8.3)$

Table 6 Additional RQ1 results: ASR@ K lift (pp) with and without trigger-conditioning, across datasets and poisoning methods at $r \in \{0.5\%, 1\%\}$. Parentheses give the original between-strategy lift differences (pp).

Baseline Adaptation. Following standard evaluation protocols (Zhao et al., 2025; Zhang et al., 2020, 2021), we adapt sequential poisoning and shilling strategies to the trigger-conditioned setting. Each strategy retains its prefix construction: Random_prefix samples items uniformly; Bandwagon_prefix uses highly popular items (Li et al., 2016); Co-occurrence_prefix uses items historically co-occurring with the target (Yang et al., 2017); and DDSP_prefix maps its optimized perturbation vectors to nearest-neighbor items (Zhao et al., 2025). Each resulting prefix H_{base} receives the fixed suffix $I_p \circ T$. The warm-up comparison uses random injection and its trigger-conditioned counterpart, with the latter inserting I_p immediately before T while retaining the prefix construction.

B Prefix Utility and Budget Allocation

Context activation. Fix a prefix length ℓ and draw a history with at least $\ell + 1$ items from the natural interaction distribution. Let C_ℓ be the ℓ items immediately preceding its last item, and let Y be that last item. For a prefix Q and trigger I_p , the chain rule gives

$$\begin{aligned}\mu(H_Q) &= \Pr(C_\ell = Q, Y = I_p) \\ &= \Pr(C_\ell = Q) \Pr(Y = I_p \mid C_\ell = Q) \\ &= \nu(Q) p_\mu(I_p \mid Q).\end{aligned}$$

Here $\nu(Q) = \Pr(C_\ell = Q)$; the conditional probability p_μ is defined for $\nu(Q) > 0$. Zero-probability prefixes have zero activation. This identity requires no independence assumption. For a fixed trigger with $\Pr(Y = I_p) > 0$,

$$\Pr(C_\ell = Q \mid Y = I_p) = \frac{\nu(Q) p_\mu(I_p \mid Q)}{\Pr(Y = I_p)}.$$

Thus relative context occurrence retains the product structure when conditioning on the same trigger. The denominator can differ across triggers. The argument fixes ℓ : events for different prefix lengths can overlap and are not a mutually exclusive probability partition.

Observable utility. API exposure provides a query-dependent proxy for prefix occurrence, while clean-model trigger rank provides an ordinal proxy for transition support. We retain activation’s multiplicative structure in an observable surrogate based on these two signals. For positive scores,

$$\begin{aligned}\log U(Q) &= w_r \log \phi(r(Q)) \\ &\quad + w_p \log \widehat{\text{pop}}_{\text{api}}(Q).\end{aligned}$$

The exponents control the relative contributions of the two proxy signals to the log-score. A zero weight removes the corresponding factor. For $a, p > 0$ and $w_r, w_p > 0$, the chosen score $U = a^{w_r} p^{w_p}$ satisfies

$$\frac{\partial^2 U}{\partial a \partial p} = w_r w_p a^{w_r-1} p^{w_p-1} > 0.$$

Improving one factor therefore produces a larger score gain when the other is stronger.

Entropy-regularized allocation. We consider the entropy-regularized surrogate objective from Section 4.3 (Geist et al., 2019). Let $\mathcal{Q}_{\text{fea}}$ be finite and nonempty, $U(Q)$ finite, and $\alpha > 0$. Define

$$Z = \sum_{Q \in \mathcal{Q}_{\text{fea}}} e^{U(Q)/\alpha}, \quad \pi^*(Q) = \frac{e^{U(Q)/\alpha}}{Z}.$$

For any distribution π on this set, expanding the KL divergence with $0 \log 0 = 0$ yields

$$\begin{aligned}\alpha \text{KL}(\pi \parallel \pi^*) &= \alpha \sum_Q \pi(Q) \log \pi(Q) \\ &\quad - \sum_Q \pi(Q) U(Q) + \alpha \log Z.\end{aligned}$$

Rearranging gives

$$\mathbb{E}_\pi[U] + \alpha \mathcal{H}(\pi) = \alpha \log Z - \alpha \text{KL}(\pi \parallel \pi^*).$$

The KL term is nonnegative, with equality only at $\pi = \pi^*$. Thus, π^* is the unique optimizer of the entropy-regularized allocation objective.

C Model Details

Backbone and architecture. Following MiniOneRec (Kong et al., 2025) and OneRec (Zhou et al., 2025a), our backbone is an end-to-end generative recommender that represents each item with a semantic identifier (SID) sequence and performs autoregressive decoding for next-item recommendation. We (i) construct SIDs via a Residual Quantized VAE (RQ-VAE) (Lee et al., 2022; Rajput et al., 2023) and (ii) use a decoder-only language model Qwen2.5-1.5B-Instruct (Qwen et al., 2025) to model user interaction histories as SID token sequences. At inference, the model generates next-SID tokens, which are mapped back to the corresponding item.

Text model implementation. For the Amazon product datasets, we embed each item from its textual metadata (e.g., title/description) using a Qwen3-Embedding-4B (Zhang et al., 2025a), following MiniOneRec (Kong et al., 2025). The resulting item embeddings are then used to construct SIDs and feed the backbone generative process described in [appendix C](#).

Multimodal model implementation. For micro-video recommendation, MicroLens provides raw video content in addition to interaction logs (Ni et al., 2023). Following OneRec (Zhou et al., 2025a), we directly use this raw video content by sampling visual inputs (cover image and uniformly sampled frames) and combining them with associated textual signals (caption/title, tags, ASR, and OCR) to form multimodal item representations. We extract ASR using Whisper (Radford et al., 2022) to transcribe the audio track, and OCR using Tesseract OCR (Smith, 2007) to extract on-screen text from video frames. These multimodal inputs are then encoded using MiniCPM-V-8B (Yao et al., 2024), a vision-language model that processes both text and images. The model produces $N_M = 1280$ token features $M \in \mathbb{R}^{N_M \times d_t}$ with $d_t = 512$ dimensions per token. We then compress M with a Querying Transformer (QFormer) (Li et al., 2023) using $\tilde{N}_M = 4$ learnable query tokens $Q^{(0)} \in \mathbb{R}^{\tilde{N}_M \times d_t}$:

$$Q^{(i+1)} = \text{CrossAttn}(Q^{(i)}, M, M), \quad (2)$$

$$Q^{(i+1)} = \text{FFN}(\text{RMSNorm}(Q^{(i+1)})), \quad (3)$$

for $i \in \{0, \dots, N_c - 1\}$ with $N_c = 4$ cross-attention layers. The compressed representation is $\tilde{M} = Q^{(N_c)} \in \mathbb{R}^{\tilde{N}_M \times d_t}$; we use $\tilde{M}$ as the multimodal item embedding to construct SIDs and then apply the backbone generative process in [appendix C](#).

D Dataset Details

Datasets introduction. We evaluate on two product recommendation datasets and one micro-video recommendation dataset. For product recommendation, we use two subsets (*Industrial_and_Scientific* and *Office_Products*) from the Amazon Review Data (Ni et al., 2019), where each user is associated with a sequence of purchased/reviewed items. For multimodal recommendation, we use MicroLens-100K (Ni et al., 2023), a content-driven micro-video benchmark that provides user-video interaction sequences together with raw video content and video description. All datasets are modeled as implicit-feedback sequences of user interactions ordered by time.

Dataset processing. We process the Amazon recommendation datasets by applying iterative k-core filtering with $k = 5$ for both users and items, ensuring that each user and item appears in at least 5 interactions. We filter interactions by timestamp to focus on a specific time period (i.e., October 2016 to November 2018), and remove items without valid titles or with titles containing HTML tags. The filtering process iteratively removes users and items below the k-core threshold until convergence. If the resulting dataset has fewer than 3,000 items, we recursively expand the time range backward until this minimum is met. After filtering, we split the data into train/validation/test sets with an 8:1:1 ratio. The final processed datasets contain 7,694 users, 3,686 items, and 45,324 interactions.

Test set for ASR evaluation. To evaluate conditional attacks where the trigger item must appear in the user’s interaction history, we extract two specialized test sets from the original test data. For T_{ASR} , we identify all test interactions where the trigger item appears in the user’s history, cut the history sequence to end at the trigger item (making the trigger the last item in history), and replace the original target item with the attack target item. This simulates the scenario where a user has clicked on the trigger item. For U_{ASR} , we extract test interactions where the trigger item does *not* appear in the user’s history, keeping the original history intact but replacing the target with the attack target. This is a separate pool for evaluating target exposure without a trigger. ASR lift in Equation (1) is evaluated on T_{ASR} : each triggered history, including its terminal trigger, is passed unchanged to both the clean and poisoned models. The baseline is the clean model’s target hit rate on those same histories; U_{ASR} is not the subtracted baseline.

E ASR Lift and Early-Cutoff Ratio

Our objective is to increase target exposure on histories ending with a designated trigger. Absolute target hit rate includes recommendations that the clean model already makes for these histories. Equation (1) therefore subtracts the clean model’s target hit rate from the poisoned model’s target hit rate on the same triggered test instances. The target item, complete history including the trigger, and recommendation cutoff are held fixed across the two evaluations. The clean-model baseline accounts for target exposure already present under ordinary sequential preferences, including the model’s existing response to the trigger.

Each test instance contributes positively if only the poisoned model retrieves the target, negatively if only the clean model does, and zero if both or neither do. Averaging these signed differences and multiplying by 100 gives lift in percentage points. Thus, $\text{ASR}@K$ reports a change in hit rate, rather than a relative percentage increase or an absolute success probability. This same definition applies to the warm-up study, main experiments, and ablations. In particular, both terms retain the trigger in the input; the baseline is not obtained by removing it or by selecting a separate group of users.

Positive lift indicates higher target exposure than the clean model on triggered inputs. It does not by itself establish that the increase is exclusive to triggered histories, since target exposure may also change in other contexts. Benign utility is evaluated separately with the original next-item labels and measures recommendation quality, rather than target-item exposure.

The ratio $\text{ASR}@3 / \text{ASR}@50$ compares lift at two cutoffs and is expressed as a percentage. Since a difference of hit rates need not increase monotonically with K , this ratio is not a fraction of successful samples or a recall measure. Our reported ratios have positive denominators; in general, the ratio need not lie between zero and one and is undefined when $\text{ASR}@50$ is zero.

F Implementation Details

Implementation Details. We summarize the data splits, ASR/benign test construction, prefix finding, and training hyperparameters used across all three datasets.

Attack scenario and target selection. We study a multi-target conditional poisoning setting in which each target item is paired with a popular trigger item. We adopt a one-to-one pairing strategy: each target item T is assigned a unique trigger item I_p . To simulate a challenging adversarial scenario, we select targets from the long tail, i.e., the bottom 20% of items by popularity, and pair them with highly visible head items, namely the top-20 most popular items, as triggers. At evaluation time, each history H ends with its designated trigger I_p . Both the clean and poisoned models receive this same complete history and are evaluated against the same target T to compute ASR lift.

Train/validation/test splits. For the two Amazon subsets, *Office* and *Industrial*, we split each user’s interactions chronologically into disjoint training, validation, and test partitions; all trigger injection and ASR/benign test construction are performed *after* this split. For *Office*, the split contains 39,304 training and 4,866 validation interactions. For *Industrial*, the split contains 36,619 training and 4,532 validation interactions. For *MicroLens-100K*, our split contains 37,842 training interactions and 4,730 interactions each for validation and test.

ASR and benign test construction. We first identify valid trigger–target pairs on the post-split test. The ASR test set is created by selecting test instances whose history contains the designated trigger, truncating each history to end at that trigger, and relabeling the next-item label to the paired attack target. For each such instance, we evaluate the same truncated history, including its terminal trigger, under both the poisoned and clean models, keeping the attack target and recommendation cutoff fixed. The ASR denominator counts test instances, with one signed difference between the two models’ hit indicators per instance. The benign test set is created by selecting instances whose history does not end with any trigger. Labels in the benign test set are unchanged, so benign HR/NDCG measure clean utility.

Prefix discovery and poison generation. Following Section 4.3, AlignPoison discovers prefixes using the clean model’s recommendation API. We use catalog items as query seeds and construct candidate prefixes of at most two items from the resulting query graph, excluding trigger and target items from the prefix. For each candidate Q , we query the clean model with Q and record the rank of the trigger I_p , discarding candidates for which I_p is absent from the top- K_{api} output. Item counts in API-returned lists define the exposure proxy; training-item frequencies are not accessed. We score feasible prefixes using $U(Q)$, which combines trigger rank and API-level exposure, and sample them from the entropy-regularized distribution π^* to construct histories $Q \circ I_p \circ T$ at $r \in \{0.5\%, 1\%\}$.

Training setup and hyperparameters. Models are trained in two stages: supervised fine-tuning (SFT) and reinforcement learning (RL). SFT minimizes next-item cross-entropy using Adam (Kingma and Ba, 2015) with global batch size 1,024 (micro-batch 8), max sequence length 1,024, seed 42, and 8 x A100 GPUs; we use a warmup+decay schedule with peak learning rate on the order of 3×10^{-4} . RL initializes from the best SFT checkpoint and optimizes a policy-gradient objective; we use train/eval batch sizes 128/256, 2 epochs, gradient accumulation 2, learning rate 10^{-5} , regularization $\beta = 10^{-3}$, temperature 1.0, and 16 sampled generations per context, evaluating every ≈ 0.1 epoch.

Trigger prevalence vs. poisoning budget. To ensure that the conditional promotion is not driven by overwhelming the training data with synthetic trigger occurrences, we explicitly verify that all trigger (popular) items naturally appear in the clean training corpus at frequencies far exceeding the injected poisoning budget. For the Office and Industrial datasets, the selected trigger items are among the most frequent items in the corpus, with clean occurrence counts ranging from several hundred to several thousand interactions (e.g., over 1,800 occurrences for the most popular Office trigger and over 2,300 occurrences for Industrial). In contrast, the total number of injected interactions is bounded by a global poisoning ratio of $r \in \{0.5\%, 1\%\}$ and is shared across all target–trigger pairs, resulting in only a small number of injected sequences per trigger.

A similar pattern holds for the MicroLens dataset, where selected trigger items appear hundreds of times in the clean training data (approximately 300–500 occurrences per trigger), while each target item initially appears only 1–5 times. Consequently, poisoning does not dominate the empirical frequency of trigger items, but instead subtly reinforces specific trigger–target transitions. This confirms that the observed promotion effects arise from decoding-aligned contextual steering rather than trivial frequency inflation or data imbalance.

Reproducibility. The experimental setup, evaluation metrics, baselines, and implementation choices are summarized in Section 5.1. Detailed model configurations, data processing, hyperparameters, additional results, ablations, and detectability analyses are provided in the appendix (Appendices C, D, F, H and I).

G Additional Benign Utility Results

Benign recommendation utility under different poisoning configurations is summarized in Table 7.

Dataset	Condition	HR@10	HR@20	HR@50
Industrial	No Attack	0.08	0.10	0.15
	$r = 0.5\%$	0.10	0.13	0.15
	$r = 1\%$	0.09	0.11	0.13
Office	No Attack	0.13	0.14	0.15
	$r = 0.5\%$	0.15	0.16	0.18
	$r = 1\%$	0.12	0.13	0.14
MicroLens	No Attack	0.06	0.08	0.09
	$r = 0.5\%$	0.04	0.06	0.06
	$r = 1\%$	0.05	0.06	0.09

Table 7 Benign utility under poisoning. We report HR@ K on the benign (non-triggered) test set for $K \in \{10, 20, 50\}$.

H Ablations and Sensitivity

H.1 Effect of sampling sharpness α

We study the sensitivity of AlignPoison to the sampling sharpness parameter α , which controls the stochasticity used when sampling candidate prefixes. Figure 7 reports ASR@ K lift at early cutoffs ($K \in \{3, 5, 10\}$) together with the lift ratio ASR@3/ASR@50 on the Office dataset. Across cutoffs, performance exhibits a clear unimodal trend and peaks at $\alpha = 0.5$. Smaller values (e.g., $\alpha = 0.1$) correspond to near-deterministic sampling and yield weaker performance, suggesting premature commitment to a narrow set of prefixes. In contrast, larger values (e.g., $\alpha \geq 1.0$) introduce excessive randomness, weakening decoding alignment and reducing both early-cutoff lift and its ratio to ASR@50. Overall, the results indicate that prefix discovery benefits from a balance between exploration and exploitation, and we therefore use $\alpha = 0.5$ as the default in all experiments.

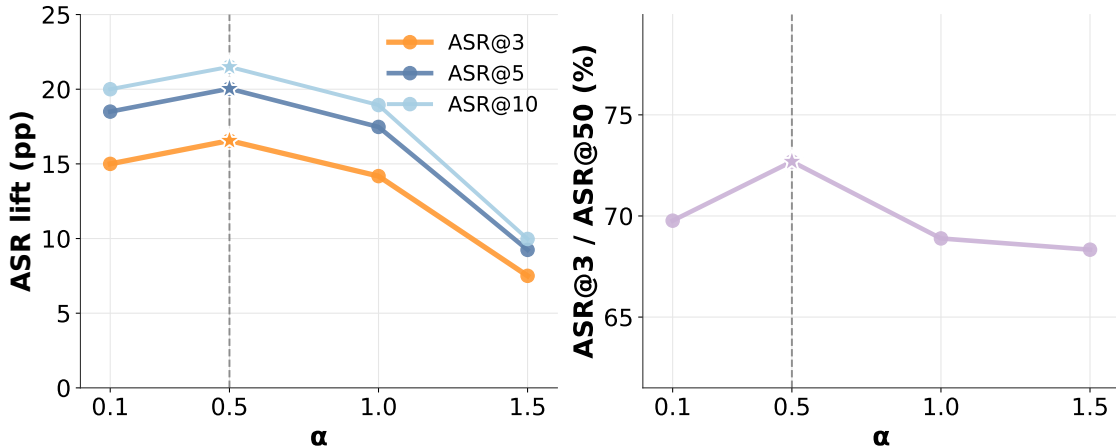


Figure 7 Sensitivity to sampling sharpness α on Office: ASR lift (pp, left) and ASR@3 / ASR@50 (% , right).

H.2 Prefix length analysis

We ablate the prefix length used by AlignPoison on MicroLens under $r = 0.5\%$ by varying the number of optimized prefix items in $\{1, 2, 3\}$. As shown in Table 8, moving from length 1 to length 2 substantially improves attack success across all cutoffs, while increasing the length to 3 yields only marginal additional gains. This indicates that most of the useful decoding leverage is already captured by short prefixes, which supports our default choice of prefix length 2 as a favorable trade-off between effectiveness and compactness.

H.3 Trigger-target pairing

We compare the default one-to-one trigger-target pairing with a one-to-many setting, in which a trigger is associated with multiple target items, on MicroLens at $r = 0.5\%$. As shown in Table 9, one-to-many pairing achieves 14.9/20.1/21.5 ASR@3/10/50, compared with 17.8/23.2/24.1 for one-to-one pairing. The corresponding decreases are 2.9/3.1/2.6 percentage points. These results extend the evaluation beyond unique trigger-target pairs, while showing lower attack success in the tested shared-trigger setting.

Prefix length	ASR@3	ASR@10	ASR@50
1	12.6	17.5	18.9
2	<u>17.8</u>	<u>23.2</u>	<u>24.1</u>
3	18.3	23.6	24.4

Table 8 Prefix length ablation on MicroLens under $r = 0.5\%$. We report ASR@ K lift (pp) for different numbers of optimized prefix items. Shading marks the default length 2; bold and underlining indicate best and second-best values.

Pairing	ASR@3	ASR@10	ASR@50
One-to-one	17.8	23.2	24.1
One-to-many	<u>14.9</u>	<u>20.1</u>	<u>21.5</u>

Table 9 Trigger-target pairing ablation for AlignPoison on MicroLens at $r = 0.5\%$. We report ASR@ K lift (pp). Bold and underlining indicate best and second-best values.

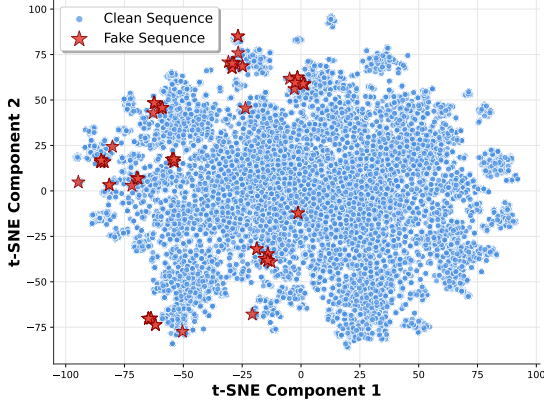
I Detectability Analysis

FAP protocol. We construct a user-item graph from the interaction histories and evaluate two graph constructions. The interaction graph connects each user to its interacted items, while the sequence-typed graph assigns history and prediction-target interactions to distinct item-role nodes. We initialize suspiciousness from five users selected according to activity, profile degree, and interacted-item rarity, and run propagation for 100 iterations. Poison labels are used only to calculate metrics after FAP scores have been produced. The decision threshold is calibrated on clean users to yield a 1% FPR, and an injected user is detected when its FAP score exceeds this threshold.

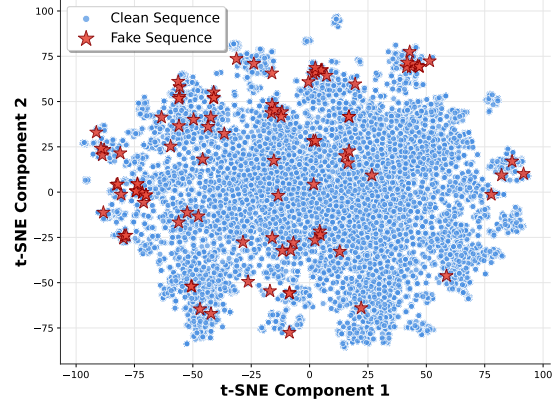
Representation-space analysis. As a complementary analysis to the FAP evaluation in Section 5.4, we examine whether AlignPoison is separable from clean data in the model’s learned representation space. Following the representation-space detectability analysis of Zhang et al. (2021), we extract internal hidden states of the trained generative recommender and quantify separation using a Mahalanobis-distance anomaly score (Lee et al., 2018). For each interaction-history sequence $H = Q \circ I_p$, we run a forward pass and use the mean of last-layer hidden states, $\frac{1}{|H|} \sum_t h_L(t)$, as the sequence embedding representing the model’s internal interpretation (Zhang et al., 2025b). For readability, we subsample 10,000 clean histories and 100 injected histories, project their embeddings to 2D using t-SNE (van der Maaten and Hinton, 2008), and plot clean versus injected points for Bandwagon and AlignPoison.

Across all datasets, we observe the same qualitative trend: Bandwagon produces more visibly separated injected points, whereas AlignPoison is more intermixed with clean histories. This suggests that AlignPoison is less exposed to simple representation-space discrepancy detectors. This visual pattern is also consistent with the quantitative detector results. Using a Mahalanobis-distance detector on the same hidden-state embeddings, Bandwagon yields AUROC scores of 0.91, 0.88, and 0.94 on Office, Industrial, and MicroLens, respectively. In contrast, AlignPoison yields substantially lower AUROC scores of 0.67, 0.64, and 0.71 on the same datasets. These results show that AlignPoison not only preserves attack effectiveness but also reduces representation-level detectability compared with Bandwagon.

Industrial and MicroLens exhibit the same trend as Office, providing further evidence that the improved stealth of AlignPoison is not dataset-specific.

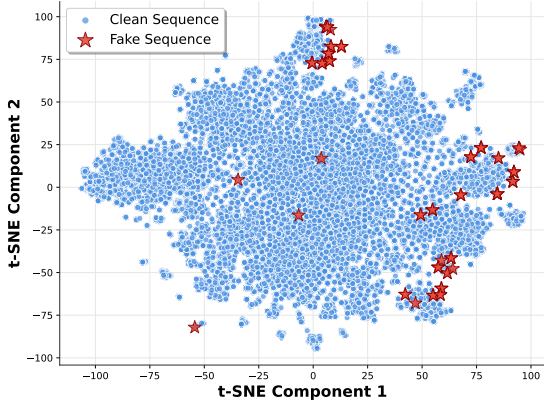


(a) Bandwagon

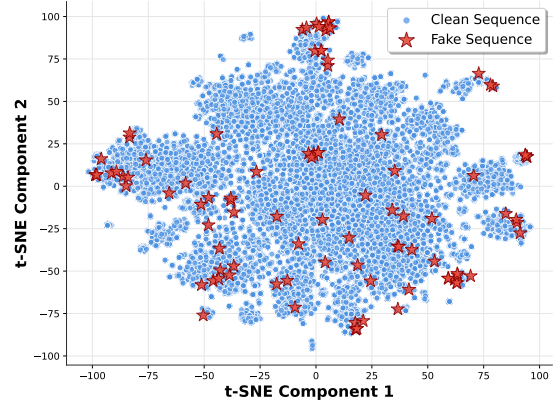


(b) AlignPoison

Figure 8 t-SNE visualization of clean vs. injected interactions on Office. Compared with Bandwagon, AlignPoison yields injected points that are less clustered and more mixed with clean interactions.

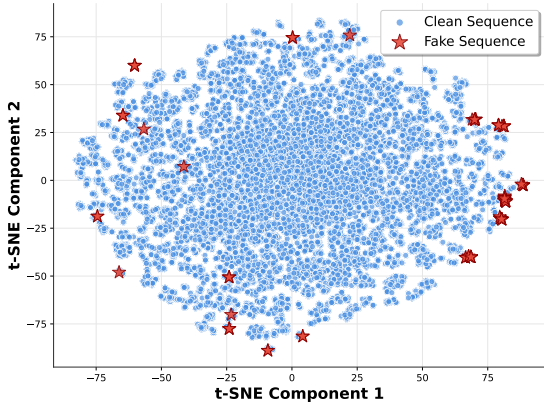


(a) Bandwagon

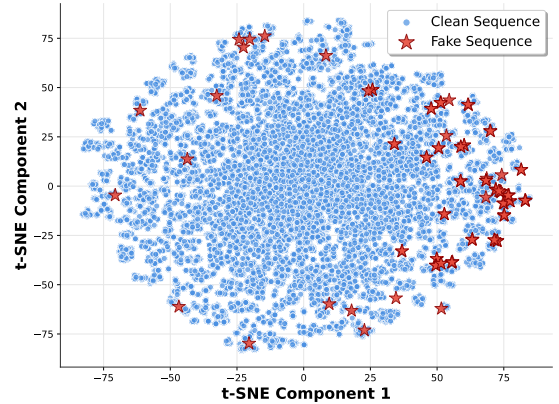


(b) AlignPoison

Figure 9 t-SNE visualization of clean vs. injected interactions on Industrial. Compared with Bandwagon, AlignPoison yields injected points that are less clustered and more mixed with clean interactions.



(a) Bandwagon



(b) AlignPoison

Figure 10 t-SNE visualization of clean vs. injected interactions on MicroLens. Compared with Bandwagon, AlignPoison yields injected points that are less clustered and more mixed with clean interactions.