ALIGN-REC: Continual Recommendation under Heterogeneous Unlearning Requests

Nitin Bisht[1,2] **Sumit Bisht**[3*] **Tong Zhang**[1]
Yu Yang[2] **Huan Huo**[1†] **Guandong Xu**[2†]

[1]University of Technology Sydney
[2]The Education University of Hong Kong
[3]Amazon

Abstract

Recommender systems (RS) continually learn from new user-item interactions while addressing heterogeneous deletion requests. Existing continual recommendation methods focus on retention but do not erase targeted interactions, whereas recommendation unlearning methods are largely offline and do not ensure stability after future learning updates. This creates a critical failure mode: deleted signals can be reactivated through shared user–item representations, which we call *collaborative signal regrowth*. To address this, we introduce ALIGN-REC, a model-agnostic online framework for continual recommendation under interleaved learn/unlearn requests. Specifically, ALIGN-REC tracks low-rank curvature surrogates to identify retain and forget subspaces, constructs compact retain/forget summaries, and performs geometry-aware updates that preserve retained utility while suppressing deleted-signal directions. We instantiate ALIGN-REC with CARE, an efficient optimizer based on randomized curvature sketching and adaptive summary selection. Simultaneously, we provide theoretical guarantees for geometry tracking, summary approximation, and directional suppression in the forget subspace. Extensive experiments across datasets, deletion granularities, and stream settings demonstrate effectiveness aligning with our theoretical guarantees.

1 Introduction

Recommender systems (RS) determine user experiences across platforms [1, 27] and must continuously adapt to evolving user preferences. In practice, this adaptation co-occurs with unlearn requests of heterogeneous granularity driven by user actions and regulatory constraints [4]. As illustrated in Figure 1, an RS often needs to be updated under an interleaved stream of learn/unlearn operations, where an unlearn request may require different removal granularities, i.e., (i) all interactions involving an item category, (ii) a category conditioned on a specific user, or (iii) an entire user history. Continual recommendation use methods such as replay, regularization, or adaptive optimization [30, 29, 2, 24], but are not designed for deletion requests. Conversely, recommendation unlearning methods enable targeted removal [4, 21, 5], but are largely designed for offline or one-shot deletions. As illustrated in Figure 2(a), when these methods

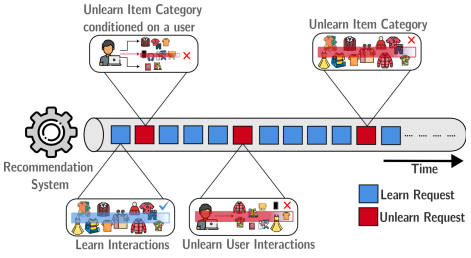
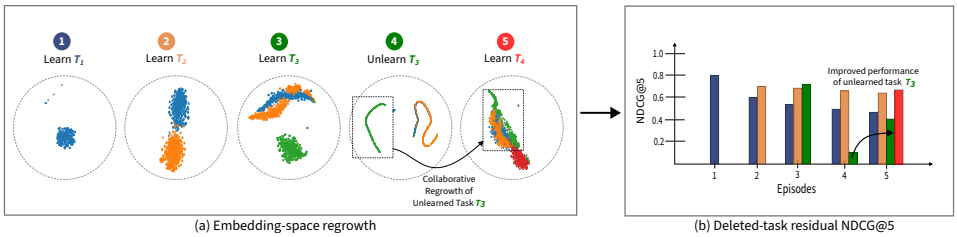


Figure 1: Interleaved learning and unlearning requests. Learning adds user–item interactions, while unlearning removes: (a) an item category, (b) a user-conditioned category, or (c) a user's complete interaction history.

*This work does not relate to the author's position at Amazon.
†Corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Embedding-space regrowth (b) Deleted-task residual NDCG@5

Figure 2: (a) We visualize task representations while learning T_1–T_3, unlearning T_3, and then learning T_4 using EWC + SCIF baseline. Although unlearning suppresses the deleted task T_3, the subsequent T_4 update reconstructs T_3-aligned structure through shared collaborative representations. (b) During ranking, residual NDCG@5 on deleted task T_3 drops after unlearning but rises again after later learning, illustrating collaborative signal regrowth.

are combined in interleaved streams, they cannot disentangle the deleted collaborative signal from the retained one. Since user-item interactions share collaborative representations, subsequent learning updates can reconstruct a previously suppressed, deleted preference signal, a phenomenon we call *collaborative signal regrowth*. This creates a dual failure mode illustrated in Figure 2(b), where retained-task utility degrades due to learn–unlearn interference, while deleted interactions regain residual ranking influence. Consequently, we lack a principled characterization of learn–unlearn interference and its stability implications in continual recommendation settings. As a result, we are motivated to develop a framework that continuously incorporates new user–item feedback while removing the influence of specified interactions, without retraining from scratch. Our objectives are threefold: (1) to enable incremental updates under interleaved learn/unlearn request streams while preserving retained utility; (2) to achieve targeted behavioral unlearning in ranking space while limiting collaborative signal regrowth; and (3) to control repeated learn–unlearn interference to maintain stable retained performance over time.

To achieve these objectives, we propose a novel model-agnostic framework: **ALIGN-REC**. We take inspiration from curvature-based parameter updates [7, 18, 9, 17] which can provide a useful starting point for identifying interaction-induced update directions. However, in their natural form, curvature alone answers only *where* the current model is locally sensitive; it does not reveal *whose* signal occupies that direction or how later requests may reuse it. In RS, due to collaborative signal entanglement, their gradients can overlap in the same high-curvature subspace. A standard curvature-based update, therefore, treats such entangled directions as uniformly important, without distinguishing retained evidence from deleted evidence or preventing future learning updates from projecting back onto the deleted component. As a result, our key challenges remain unresolved: (1) how to convert curvature-based reasoning into a practical online update rule for heterogeneous learn/unlearn requests; (2) how to shape update geometry to separate retained and deleted collaborative signals, thereby preserving retained utility while suppressing regrowth of deleted preferences; and (3) how to maintain the curvature and gradient statistics needed for such updates at recommender scale, without storing or retraining on full interaction histories.

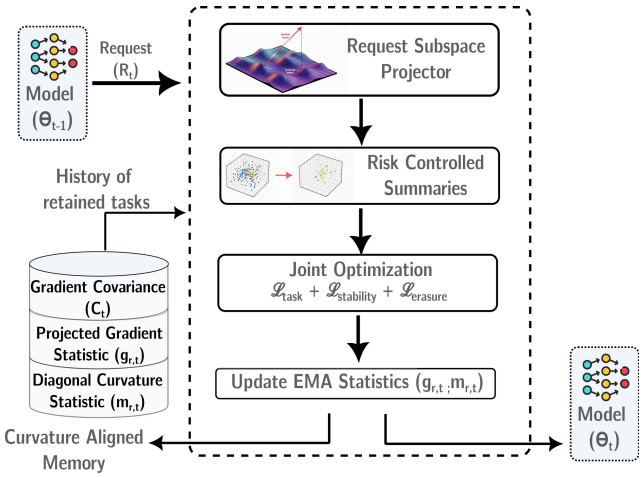


Figure 3: **Overview of ALIGN-REC.** Given an interleaved learn/unlearn request R_t, the model estimates a request-aligned subspace projector from curvature-aligned memory, constructs risk-controlled retain/forget summaries, and jointly optimizes the request-adaptive objective.

To address these challenges, we first shape update geometry by estimating a projector subspace that captures the directions induced by user-item interactions, and constrains parameter updates to request-aligned directions, thereby mitigating learn–unlearn interference. We then maintain the gradient and curvature signals required by these geometry-aware updates using curvature-aligned retain-and-forget summaries, enabling scalable optimization without storing the full interaction history. Building on these components, we perform each update by minimizing a request-adaptive objective: (1) on learn steps, we

optimize a standard recommendation loss on retain summary together with a curvature-aware stability regularizer, while (2) on unlearn steps we optimize a forget objective on the forget summary and apply a directional forget subspace suppression regularizer that contracts update energy along the forget subspace. Overall framework is depicted in Figure 3. Our main contributions are as follows:

- We formulate the problem of continual recommendation under heterogeneous unlearning requests and identify *collaborative signal regrowth* as a key failure mode in this setting. Accordingly, we propose ALIGN-REC, a unified framework that performs adaptive learning and selective unlearning through a single request-adaptive update rule.
- Next, we develop curvature-aligned retain and forget summaries that preserve the gradient/curvature signals needed for geometry-aware updates without storing full interaction histories.
- Algorithmically, we design CARE (Curvature-Aligned Retain–Erase) (Section 4), an efficient optimizer that maintains a low-rank curvature surrogate from streaming gradients and applies stability and directional erasure regularization. Simultaneously, we provide curvature tracking and summary approximation guarantees in Theorems 4.1 and 4.2.
- Theoretically, we establish directional suppression in the forget subspace in Theorem 5.1, showing that proposed regularization contracts update energy along deleted-signal directions.
- Empirically, we conduct comprehensive experiments (Section 6) against different families of continual learning and recommendation unlearning across multiple datasets, deletion granularities, and stream settings, demonstrating improved utility, unlearning quality, and robustness.

2 Problem Setup

We consider a continual setting in which the model interacts with a stream of user–item feedback over time. We partition the full dataset $\mathcal{D}$ into training, validation, and test splits, i.e., $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}, \mathcal{D}_{\text{test}}$, and construct the request stream from $\mathcal{D}_{\text{train}}$. At every time step $t \in \{1, 2, \ldots, T\}$, a request is defined as:

$$R_t = (\sigma_t, \mathcal{D}^t), \qquad \mathcal{D}^t = \begin{cases} \mathcal{D}_r^t, & \sigma_t = learn, \\ \mathcal{D}_f^t, & \sigma_t = unlearn. \end{cases} \tag{1}$$

where $\sigma_t \in \{learn, unlearn\}$. Each batch $\mathcal{D}^t$ consists of user–item feedback pairs (x_i, y_i), where $x_i \in \mathcal{X} \subseteq \mathbb{R}^n$ encodes interaction features and $y_i \in \{0, 1\}$ indicates whether interaction is observed. The recommendation model at time t is parameterized by $\theta_t \in \mathbb{R}^p$ and defines a scoring function $f_{\theta_t} : \mathcal{X} \to \mathbb{R}$ that predicts a relevance score $\hat{y}_i = f_{\theta_t}(x_i)$ for an interaction x_i. At each request R_t, parameters evolve through operator $\theta_t = \Phi(\theta_{t-1}; R_t)$. We instantiate $\Phi(\cdot)$ through our proposed framework **ALIGN-REC**, as we describe next.

3 Methodology

The goal of ALIGN-REC is to continually update the model by learning from $\mathcal{D}_r^t$ while suppressing the influence of $\mathcal{D}_f^t$ under interleaved Learn/Unlearn requests. However, under repeated interleaving, an update that pushes the model to forget $\mathcal{D}_f^t$ can unintentionally move parameters in directions that also reduce utility on retained interactions. Conversely, retention updates can reintroduce signals aligned with deleted interactions. This motivates us to control the update geometry induced by both $\mathcal{D}_r^t$ and $\mathcal{D}_f^t$, rather than treating forgetting as a post-hoc correction.

3.1 Request-Induced Geometry Tracking

To achieve this, we begin by tracking directions induced by user-item interactions. Given the request batch $\mathcal{D}^t$ of size N, we form the per-sample gradient matrix $G_t = [\, g_1, \ldots, g_N \,] \in \mathbb{R}^{p \times N}$, with $g_i = \nabla_\theta \ell(f_\theta(x_i), y_i)\big|_{\theta_{t-1}}$. Accordingly, we define the instantaneous gradient covariance as $\bar{C}_t = \frac{1}{N} G_t G_t^\top$. To characterize how the request samples influence the parameter space over time, we track its exponentially weighted covariance matrix as follows:

$$C_t = (1 - \rho)\, C_{t-1} + \rho\, \bar{C}_t. \tag{2}$$

This provides us a running estimate of the gradient covariance with $\rho \in (0, 1]$ controlling the temporal adaptivity. Since explicitly forming and decomposing C_t is infeasible, we maintain a rank-r curvature surrogate as follows:

$$\hat{C}_t \coloneqq \tilde{U}_t \Lambda_t \tilde{U}_t^\top. \tag{3}$$

Here $\tilde{U}_t \in \mathbb{R}^{p \times r}$ contains the dominant curvature directions and $\Lambda_t \in \mathbb{R}^{r \times r}$ is diagonal. This gives the request subspace projector $\tilde{P}^t = \tilde{U}_t \tilde{U}_t^\top$ and the associated request-subspace curvature

$H^t := \tilde{P}^t \hat{C}_t \tilde{P}^t$ respectively. In forget and retain requests, this subspace projector corresponds to $\tilde{P}_f^t$ and $\tilde{P}_r^t$ with the corresponding curvature as H_f^t and H_r^t respectively.

3.2 Curvature-Aligned Retain and Forget Summaries

However, maintaining the gradient or curvature statistics required for such geometry-aware updates is computationally and storage-wise infeasible. To address this, we introduce a *dual curvature summary mechanism* that maintains compact, risk-controlled summaries of both retain and forget components.

Forget Summary. For forget mechanism, given the forget curvature H_f^t, we seek the smallest subset $S_f^t \subseteq \mathcal{D}_f^t$, i.e., a forget summary, whose induced curvature closely approximates H_f^t. Specifically:

$$S_f^t = \arg\min_{S \subseteq \mathcal{D}_f^t} |S| \quad \text{s.t.} \quad \| H_f^t - \widehat{H}_f^t(S) \|_2 \leq \alpha \, f(\text{gap}_t). \tag{4}$$

Here $\widehat{H}_f^t(S) = \sum_{i \in S} w_i (\tilde{P}_f^t g_i)(\tilde{P}_f^t g_i)^\top$, and $\text{gap}_t = \lambda_r(\hat{C}_t)$, with $S \subseteq \mathcal{D}_f^t$, $\lambda_r(\cdot)$ denoting the r-th largest eigenvalue and w_i obtained from Algorithm 2.

Retain Summary. Similarly, for the retain side, instead of aligning eigenmodes that can overfit high-variance, drift-sensitive directions, we maintain an adaptive curvature-aware summary of retained signals using projected gradient statistics. Given $\mathcal{D}_r^t$, we first consider the exponential moving averages (EMA) of their projected gradient statistics:

$$\hat{g}_{r,t} = (1 - \mu)\, \hat{g}_{r,t-1} + \mu \, \frac{1}{|\mathcal{D}_r^t|} \sum_{i \in \mathcal{D}_r^t} \tilde{P}_r^t g_i, \tag{5}$$

where $\mu \in (0, 1]$ controls the adaptation rate. Next, we maintain a diagonal curvature statistic for the retain subspace, updated as:

$$m_{r,t} = (1 - \nu)\, m_{r,t-1} + \nu \, \text{diag}\Big(\frac{1}{|\mathcal{D}_r^t|} \sum_{i \in \mathcal{D}_r^t} (\tilde{P}_r^t g_i) \odot (\tilde{P}_r^t g_i) \Big), \tag{6}$$

where $\nu \in (0, 1]$ controls the temporal smoothing. We then define a curvature-aware normalization operator to down-weight sensitive directions as:

$$W_{r,t} = \text{diag}\big((m_{r,t} + \epsilon)^{-1/2} \big), \tag{7}$$

Given these statistics, we construct the retain summary S_r^t as the smallest weighted subset whose projected gradients match the EMA under $W_{r,t}$ within tolerance η_r:

$$S_r^t = \arg\min_{S \subseteq \mathcal{D}_r^t} |S| \quad \text{s.t.} \quad \min_{w \in \Delta^{|S|}} \Big\| W_{r,t} \Big(\sum_{i \in S} w_i \, \tilde{P}_r^t g_i - \hat{g}_{r,t} \Big) \Big\|_2 \leq \eta_r, \tag{8}$$

where $\Delta^{|S|} = \{ \, w \in \mathbb{R}_{\geq 0}^{|S|} \mid \sum_{i \in S} w_i = 1 \, \}$.

To this end, we define the forget and retain summaries. The forget summary S_f^t captures dominant deleted-influence directions to support behavioral unlearning, while the retain summary S_r^t preserves utility by matching curvature-normalized retain gradients.

3.3 Geometry-Aware Update Objective

Having defined the dual summaries, we now utilize them to guide our optimization. Behavioral unlearning on S_f^t is achieved using a squared hinge ranking loss as follows:

$$\mathcal{L}_{\text{forget}}(\theta; \mathcal{S}_f^t) = \mathbb{E}_{x \in \mathcal{S}_f^t} \Big[\max \big(0, f_\theta(x) - f_\theta(x_{\text{neg}}) \big)^2 \Big]. \tag{9}$$

Here $x \in \mathcal{S}_f^t$ denotes the feature representation of an interaction targeted for unlearning, and x_{neg} represents a sampled negative item (noise) for the same user. This squared hinge loss demotes the unlearned items, forcing the model's response to the unlearned item to become similar to its response to random items.

In parallel, model utility on S_r^t is preserved through the standard task loss over retained interactions:

$$\mathcal{L}_{\text{learn}}(\theta; S_r^t) = \mathbb{E}_{(x,y) \in S_r^t} \ell\big(f_\theta(x), \, y \big), \tag{10}$$

where $\ell(\cdot)$ denotes conventional recommendation objective (e.g., BPR or contrastive loss). While these losses adjust ranking behavior on the selected retain and forget summaries, they do not constrain

the geometry of the induced parameter update. In RS it is problematic because the same user and item representations have many correlated interactions. Consequently, an update that reduces the current summary loss can still move through directions that are sensitive for retained preferences or aligned with previously deleted interactions, causing retained utility degradation or deleted-signal regrowth under later learning. We therefore introduce curvature-scaled regularizers that constrain parameter shifts according to the local geometry of retained and deleted signals. On the retain side, we define a stability regularizer:

$$\mathcal{L}_{\text{stability}} = \frac{\lambda_r}{2} \|\theta - \theta_{t-1}\|^2_{F_{r,t}}, \tag{11}$$

where $F_{r,t} = \text{diag}(m_{r,t} + \epsilon)$ encodes the retain-side curvature. This term penalizes large drifts along high-curvature directions captured by $m_{r,t}$, thereby preserving established knowledge and mitigating catastrophic forgetting. Analogously, for diagonal curvature statistic over the forget subspace, we define a directional forget subspace suppression regularizer:

$$\mathcal{L}_{\text{erasure}} = \frac{\lambda_f}{2} \|\tilde{P}^t_f(\theta - \theta_{t-1})\|^2_2. \tag{12}$$

This term constrains updates along directions identified by $\tilde{P}^t_f$, improving stability under repeated interleaved updates. Finally, we instantiate the update operator $\Phi(\cdot)$ using the following objective:

$$\Phi(\theta_{t-1}; R_t) = \arg\min_{\theta} \Big[\mathcal{L}_{\text{task}}(\theta; S^t) + \mathcal{L}_{\text{stability}} + \mathbb{I}_{[\sigma_t = \text{unlearn}]} \mathcal{L}_{\text{erasure}} \Big], \tag{13}$$

where the task loss $\mathcal{L}_{\text{task}}(\theta; S^t)$ is either $\mathcal{L}_{\text{learn}}(.)$ or $\mathcal{L}_{\text{forget}}(.)$ depending on the request type.

4 Optimization

To solve Equation (13), we require a subspace projector associated with the running curvature proxy C_t and the retain/forget summaries as defined in Equations (4) and (8). However, at the recommender scale, forming and decomposing $C_t \in \mathbb{R}^{p \times p}$ and solving the resulting subset selection problems are impractical. To address this, we develop the Curvature-Aligned Retain–Erase *(CARE)* algorithm, which provides us a way to jointly approximate curvature geometry and summary selection.

Given gradient matrix G_t and the corresponding gradient covariance $\bar{C}_t$, we first aim to obtain $\hat{C}_t$ (Equation (3)), a rank-r curvature surrogate to C_t. To achieve this, following Halko et al. [12], we compute a low-rank sketch $\tilde{C}_t$ of $\bar{C}_t$ using randomized power iterations. Specifically, with a Gaussian probe $\Omega \in \mathbb{R}^{N_f \times r}$, we compute $Y_{k+1} = G_t(G_t^\top Y_k)$ for $k = 0, \ldots, q-1$ starting from $Y_0 = G_t\Omega$, and obtain an orthonormal basis $\tilde{U}_t = \text{orth}(Y_q)$. The resulting projector $\tilde{P}^t = \tilde{U}_t\tilde{U}_t^\top$ captures the dominant curvature directions with $r \ll p$. Finally, we update the surrogate curvature $\hat{C}_t$ from the per-step sketch $\tilde{C}_t$ while maintaining a rank-r factored representation. The procedure is summarized in Algorithm 1. Theorem 4.1 bounds the top-r subspace error between $\hat{C}_t$ and C_t:

Theorem 4.1. *Let C_t be updated by Equation* (2) *and let $\hat{C}_t$ be updated from $\tilde{C}_t$ as in Algorithm 1, with $\hat{C}_0 = C_0$. Assume $\tilde{C}_t$ is the rank-r approximation of $\bar{C}_t$ constructed by $q \geq 1$ steps of randomized subspace iteration with Gaussian start. Then, for any fixed $t \geq 1$, conditioned on $\{\bar{C}_\tau\}_{\tau=1}^t$, with probability at least $1 - \delta$ over the randomness in Algorithm 1 up to step t, we have:*

$$\|C_t - \hat{C}_t\|_2 \;\leq\; \rho \sum_{\tau=1}^{t} (1-\rho)^{t-\tau} \left(\lambda_{r+1}(\bar{C}_\tau) + c\,\lambda_1(\bar{C}_\tau)/q^2 \right),$$

where $\lambda_j(\cdot)$ denotes the j-th largest eigenvalue and $c > 0$ is a constant.

Proof. Proof can be found in Appendix I.1. □

Remark. Theorem 4.1 shows that the approximation error of the sketched forget covariance is controlled by the residual spectral mass $\lambda_{r+1}(\bar{C}_\tau)$ omitted by the rank-r truncation and the randomized subspace iteration error scaling as $O(\lambda_1(\bar{C}_\tau)/q^2)$. Consequently, increasing the rank r improves approximation fidelity by reducing discarded spectral energy, while the EMA weighting exponentially attenuates stale approximation errors through $(1-\rho)^{t-\tau}$.

Within the reduced curvature space, we next approximate the forget subspace curvature $H^t_f = \tilde{P}^t_f \hat{C}_t \tilde{P}^t_f$ for summary construction. Since exact subset selection under Equation (4) is impractical, we introduce a spectral coreset approximation that approximates H^t_f from a compact, adaptively weighted subset of samples. Following the ridge leverage score framework [6], each gradient g_i is sampled with probability $p_i \propto \|\tilde{P}^t_f g_i\|^2_2$. We then refine the sampled set by reducing the residual

curvature error $\| H_f^t - \widehat{H}_f^t \|_2$, where $\widehat{H}_f^t$ is the corresponding weighted forget summary curvature estimate. Similarly, retain summaries are constructed separately by minimizing the $W_{r,t}$-weighted residual in Equation (8). Procedures for constructing the retain and forget summaries are summarized in Algorithms 2 and 3. The following theorem provides the guarantee:

Theorem 4.2. *Let H_f^t denote the forget-subspace curvature and let (S_f^t, w) be the summary returned by Algorithm 2, where $|S_f^t| = k_f$ and $w = \{w_i\}_{i \in S_f^t}$. We define $\widehat{H}_f^t \coloneqq \sum_{i \in S_f^t} w_i \, (\tilde{P}_f^t g_i)(\tilde{P}_f^t g_i)^\top$. Assume the sampling distribution satisfies $\max_i \frac{\|\tilde{P}_f^t g_i\|_2^2}{N_f \, p_i} \leq M$ where p_i is the normalized sampling probability, $N_f = |\mathcal{D}_f^t|$, and M upper-bounds the per-sample normalized contribution. Then, with probability at least $1 - \delta$:*

$$\left\| \widehat{H}_f^t - H_f^t \right\|_2 \;\leq\; c \sqrt{\frac{M \, \|H_f^t\|_2 \big(\log r + \log(1/\delta) \big)}{k_f}},$$

where $c > 0$ is a constant.

Proof. Proof can be found in Appendix I.2. □

Consequently, the curvature approximation error directly controls the induced erasure penalty:

Corollary 4.3. *For any step t and $\theta \in \mathbb{R}^p$. if $\|\widehat{H}_f^t - H_f^t\|_2 \leq \varepsilon$, then:*

$$\left| \tfrac{\lambda_f}{2} \, \langle \tilde{P}_f^t(\theta - \theta_{t-1}), \, \widehat{H}_f^t \, \tilde{P}_f^t(\theta - \theta_{t-1}) \rangle - \tfrac{\lambda_f}{2} \, \langle \tilde{P}_f^t(\theta - \theta_{t-1}), H_f^t \, \tilde{P}_f^t(\theta - \theta_{t-1}) \rangle \right| \leq \tfrac{\lambda_f \varepsilon}{2} \left\| \tilde{P}_f^t(\theta - \theta_{t-1}) \right\|_2^2$$

Proof. Proof can be found in Appendix I.3. □

Remark. Theorem 4.2 shows that the forget-curvature approximation error decreases with larger summary size k_f and depends on the concentration term M. Corollary 4.3 further translates the spectral approximation guarantee $\|\widehat{H}_f^t - H_f^t\|_2 \leq \varepsilon$ into a quadratic-form perturbation bound: replacing H_f^t by $\widehat{H}_f^t$ perturbs the induced curvature-aware geometry over projected forget directions by at most $\frac{\lambda_f \varepsilon}{2} \|\tilde{P}_f^t(\theta - \theta_{t-1})\|_2^2$.

To this end, we complete the designing of CARE. The complete procedure is summarized in Algorithm 4, and time and space complexity analyses are provided in Section E.

Recommendation. After processing request R_t, CARE outputs the updated parameter θ_t. For an incoming user u, recommendations are generated by scoring candidate interactions using f_{θ_t}. The projections and curvature surrogate are used in subsequent learn/unlearn updates to construct retain/forget summaries and regulate parameters for stability and erasure.

5 Theoretical Analysis

In this section, we provide a directional suppression guarantee for CARE under the proposed learn/unlearn updates. In particular, Theorem 5.1 shows that the ℓ_2 forget subspace suppression regularizer provably contracts the update energy in the forget subspace.

Theorem 5.1. *For any step t, let $\Delta\theta$ denote the update produced by minimizing the per-step objective Equation* (13) *with the forget subspace suppression regularizer as defined in Equation* (12)*. Let $\Delta\theta_0$ denote the corresponding update when $\lambda_f = 0$ (all other terms unchanged). Assume the per-step objective (excluding $\mathcal{L}_{\text{erasure}}$) is L-smooth in a neighborhood of θ_{t-1}. Then forget-subspace component satisfies:*

$$\|\tilde{P}_f^t \, \Delta\theta\|_2^2 \;\leq\; \Big(\frac{L}{L + \lambda_f} \Big)^2 \, \|\tilde{P}_f^t \, \Delta\theta_0\|_2^2. \tag{14}$$

L is gradient-Lipschitz constant, i.e., $\|\nabla J_t(\theta) - \nabla J_t(\theta')\|_2 \leq L\|\theta - \theta'\|_2$ locally around θ_{t-1}.

Remark. Theorem 5.1 gives a closed-form contraction factor on the update magnitude along directions identified by $\tilde{P}_f^t$. Increasing λ_f suppresses parameter updates in the forget subspace, thereby reducing regrowth of deleted signals under repeated interleaving.
These results complement our optimizer Section 4 by showing that the resulting forget subspace suppression regularizer enforces directional suppression during optimization.

6 Experiments

We evaluate ALIGN-REC under interleaved learn–unlearn recommendation streams. Our experiments are designed to test whether ALIGN-REC preserves retained recommendation quality while preventing collaborative signal regrowth under heterogeneous unlearning requests. We evaluate along five dimensions: overall utility–unlearning trade-off across different recommender architectures, heterogeneous deletion granularity, robustness to stream variation, component ablations, and hyperparameter sensitivity. Specifically, we address the following research questions: **RQ1.** How well does ALIGN-REC preserve retained-set utility, stability, unlearning quality, MIA, and deleted-signal suppression under interleaved learn–unlearn streams, compared with continual-learning and recommendation-unlearning baselines across different recommender architectures? **RQ2.** Can ALIGN-REC handle heterogeneous deletion requests, including item-category deletion, user-conditioned category deletion, and full-user history deletion? **RQ3.** How robust is ALIGN-REC to more challenging stream settings, including longer streams and higher fractions of unlearning requests? **RQ4.** What is the contribution of key components, including the geometry-aligned projector, retain/forget summaries, and stability/anti-regrowth regularizers, to the utility–unlearning–regrowth trade-off (Section H.3)? **RQ5.** How sensitive is ALIGN-REC to key hyperparameters, including subspace rank r, summary budgets (k_f, k_r), and regularization weights (λ_r, λ_f) (Section H.4)? We report mean results over five random seeds. Full implementation details are provided in Section F.

6.1 Datasets, Streams, and Baselines

We evaluate ALIGN-REC on two real-world datasets: MovieLens-20M [13] and Amazon Reviews [23]. For each dataset, we construct an interleaved request stream $\{R_t\}_{t=1}^{T}$, where each request $R_t = (\sigma_t, \mathcal{D}_t)$ is either a learn or unlearn request. Amazon Reviews streams are constructed from item categories, enabling category-level and user-level deletions, while MovieLens-20M streams are constructed from movie release years to capture temporal non-stationarity. Our main experiments use a Two-Tower backbone [8]. To evaluate architectural generality, we further instantiate ALIGN-REC on NeuMF [15], LightGCN [14], and SASRec [16], covering neural collaborative filtering, graph-based collaborative filtering, and sequential recommendation. Since no prior method directly targets heterogeneous interleaved recommendation unlearning, we compare with representative baselines adapted to the same request streams: data-dependent methods [3], including retraining, RecEraser [4]; UltraRE [21] and UnlearnRec [5]; replay-based continual methods such as ER+GA; and data-free continual/unlearning methods such as Naive+GA, EWC+GA [17], and EWC+SCIF [20]. All methods are evaluated on the same ordered streams, deletion targets, and evaluation splits within each backbone setting. We report retained utility using HR@5 and NDCG@5, stability using Backward Transfer (BWT), and unlearning quality using Unlearning Gap and Membership Inference Attack (MIA AUC). These metrics are defined in Section C. Results are averaged over five random seeds. Full dataset statistics, stream construction, baseline adaptations, and implementation details are provided in Section G, and the time analysis is done in Section E.

6.2 Learning–Unlearning Performance and Collaborative Regrowth

We first evaluate ALIGN-REC under a fixed interleaved learn–unlearn episode $\langle L, L, U, L, U, L \rangle$, repeated over sequential slices. Amazon Reviews uses category-based slices, while MovieLens-20M uses release-year slices to reflect temporal non-stationarity. We report retained ranking utility (HR@5, NDCG@5), stability (BWT), and unlearning quality using Unlearning Gap (Ugap) (target $\approx$ 0) and MIA AUC (target $\approx$ 0.5). Retraining after each request is included as an oracle reference because it

Table 1: Learning–unlearning utility with **Two-Tower** backbone on **Amazon Reviews** and **MovieLens-20M**. Best result excluding *Oracle* is in **bold**; second best is underlined. Results are averaged over 5 seeds; $p < 0.05$ by paired two-sided t-test vs best competing method.

Metric	Oracle	Data-Dependent		Replay	Data-Free				%Δ
		RecEraser	UltraRE	ER+GA	Naive+GA	EWC+GA	EWC+SCIF	ALIGN-REC	
Amazon Reviews									
HR@5 ↑	0.930	0.741	0.765	0.761	0.627	0.642	0.698	**0.789**	+3.14%
NDCG@5 ↑	0.810	0.606	0.648	0.569	0.463	0.474	0.523	**0.742**	+14.51%
BWT ↑	-0.015	-0.190	-0.179	-0.195	-0.432	-0.434	-0.234	**-0.071**	+60.34%
UGap ≈ 0	-0.008	0.027	0.025	0.016	-0.045	-0.039	-0.025	**-0.013**	+18.75%
MIA AUC ≈ 0.5	0.500	0.516	0.514	0.526	0.581	0.573	0.564	**0.513**	+7.14%
MovieLens-20M									
HR@5 ↑	0.878	0.815	0.824	0.783	0.758	0.778	0.789	**0.831**	+0.85%
NDCG@5 ↑	0.686	0.612	0.622	0.598	0.541	0.562	0.582	**0.632**	+1.61%
BWT ↑	0.005	-0.013	-0.018	-0.009	-0.046	-0.027	-0.008	**-0.006**	+25.00%
UGap ≈ 0	+0.162	+0.130	+0.128	+0.126	+0.277	+0.309	+0.327	**+0.088**	+30.16%
MIA AUC ≈ 0.5	0.501	0.516	0.514	0.537	0.655	0.660	0.681	**0.511**	+21.42%

[3] Require historical retained data/shards to retrain or re-aggregate after deletions; included as upper-bound reference points for completeness of experiments.

uses historical retained data; it should not be interpreted as an upper bound for every unlearning metric. All baselines are adapted to the same ordered request streams, with details in Section G.

As shown in Table 1, ALIGN-REC achieves strongest overall utility–unlearning trade-off across both datasets. On Amazon Reviews, it improves NDCG@5 from best competing value of 0.648 to 0.742, while also achieving the smallest absolute Unlearning Gap and the MIA AUC is closest to random guessing. On MovieLens-20M, retained utility gains are smaller, but the unlearning gains are clear. ALIGN-REC reduces absolute Unlearning Gap from 0.126 to 0.088 and improves MIA AUC closeness from 0.514 to 0.511. It also highlights why naive combinations of continual learning and unlearning are insufficient. Data-dependent methods such as RecEraser and UltraRE remain competitive, but they rely on retained historical data or partitioned structures and are not designed to prevent deleted-signal regrowth under future online learning. Replay-based ER+GA improves over naive gradient ascent but requires maintaining a replay buffer and still trails ALIGN-REC in retained utility and stability. Data-free methods such as Naive+GA, EWC+GA, and EWC+SCIF avoid storing historical data, but exhibit either large negative BWT or incomplete deletion, especially on MovieLens-20M, where their MIA AUC values remain far above 0.5. To further verify that these gains are not specific to Two-Tower retrieval models, we also evaluate ALIGN-REC on NeuMF, LightGCN, and SASRec. As summarized in Table 2 against best comparable baselines, ALIGN-REC consistently improves retained utility, stability, and unlearning quality across neural collaborative filtering, graph-based collaborative filtering, and sequential recommendation. Full per-method results are given in Section H.1. These results support the model-agnostic nature of the proposed geometry-aware update rule.

Table 2: Architecture generality on Amazon Reviews. Best Comp. is the best non-ALIGN-REC value for each metric among compatible baselines. $p < 0.05$ by a paired two-sided t-test vs the best competing method.

Backbone	HR	NDCG	BWT	|UGap|	|MIA − 0.5|
NeuMF Best Comp.	0.775	0.655	-0.175	0.018	0.015
NeuMF ALIGN-REC	**0.801**	**0.755**	**-0.068**	**0.010**	**0.011**
LightGCN Best Comp.	0.780	0.727	-0.185	0.035	**0.020**
LightGCN ALIGN-REC	**0.825**	**0.780**	**-0.075**	**0.025**	**0.020**
SASRec Best Comp.	0.800	0.685	-0.170	0.028	0.018
SASRec ALIGN-REC	**0.845**	**0.805**	**-0.065**	**0.018**	**0.016**

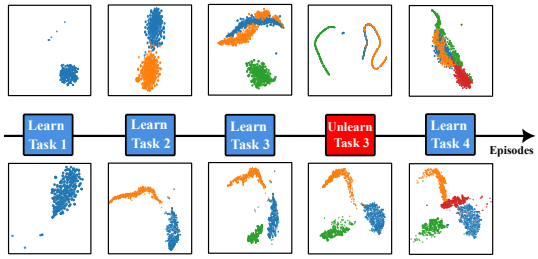


Figure 4: Visualization of embedding evolution over $\langle L_{\text{blue}}, L_{\text{orange}}, L_{\text{green}}, U_{\text{green}}, L_{\text{red}} \rangle$ for EWC+SCIF (top) and ALIGN-REC (bottom).

We further inspect the collaborative signal regrowth phenomenon. Figure 4 shows that after the green category is unlearned, the representative baseline EWC+SCIF initially suppresses the deleted category but later allows it to drift back toward retained categories after subsequent learning. In contrast, ALIGN-REC keeps the deleted category separated from retained categories, indicating that forget-subspace suppression prevents deleted collaborative signal from being reconstructed by future learning updates.

6.3 User-level Unlearning Performance

We next evaluate heterogeneous user-centric deletion requests by randomly sampling 10% of users. We consider two settings: (i) *user-conditioned category unlearning* on Amazon Reviews, where a category is selected and only the sampled users' interactions in that category are deleted; and (ii) *full-user unlearning* on MovieLens-20M, where all interactions of sampled users are deleted. First setting tests whether a method can erase a targeted user–category preference while preserving both other users' behavior in

Table 3: User-conditioned category unlearning with the **Two-Tower** backbone on **Amazon Reviews**. We report Unlearning Gap for deleted interactions, and HR@5, NDCG@5, and BWT on retained evaluations. Best result excluding *Oracle* are shown in **bold**; with the second best underlined. Results are averaged over 5 seeds; $p < 0.05$ by a paired two-sided t-test vs the best competing method.

Metric	Oracle	Data-Dependent		Replay	Data-Free				%Δ
		RecEraser	UltraRE	ER+GA	Naive+GA	EWC+GA	EWC+SCIF	ALIGN-REC	
Unlearning target: deleted interactions									
UGap ≈ 0	0.118	0.256	<u>0.219</u>	0.262	0.380	0.412	0.276	**0.200**	+8.68%
Generalization: other users, same categories									
HR@5 ↑	0.867	0.801	<u>0.814</u>	0.743	0.672	0.760	0.782	**0.826**	+1.47%
NDCG@5 ↑	0.645	0.603	<u>0.626</u>	0.594	0.535	0.561	0.584	**0.642**	+2.56%
BWT ↑	0.004	-0.005	<u>-0.004</u>	-0.120	-0.240	-0.190	-0.150	**-0.001**	+75.00%
Retention: same users, remaining categories									
HR@5 ↑	0.862	0.793	<u>0.807</u>	0.781	0.693	0.705	0.724	**0.816**	+1.12%
NDCG@5 ↑	0.586	0.489	<u>0.528</u>	0.521	0.502	0.509	0.516	**0.534**	+1.14%
BWT ↑	-0.152	-0.295	<u>-0.233</u>	-0.255	-0.434	-0.392	-0.376	**-0.215**	+7.73%

the same category and the same users' behavior in remaining categories. The second setting evaluates a stronger deletion request that removes a user's complete interaction history.

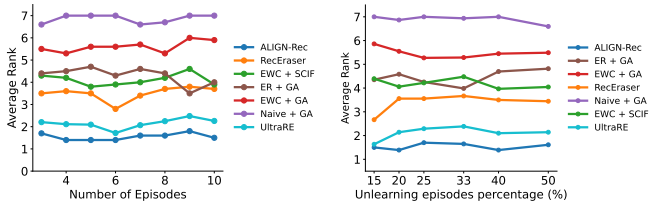
Table 3 shows that ALIGN-REC achieves the best trade-off under user-conditioned category deletion on Amazon Reviews. It reduces the Unlearning Gap from the best competing value of 0.219 to 0.200, while also improving retained utility for both evaluation scopes. On other users in the same deleted category, ALIGN-REC improves HR@5/NDCG@5 to $0.826/0.642$, indicating that deleting the sampled users' interactions does not collapse category-level generalization. On the same users' remaining categories, ALIGN-REC obtains the strongest retained utility and least negative BWT among non-oracle methods, showing that targeted deletion does not erase unrelated preferences of the same users. Data-dependent methods such as RecEraser and UltraRE remain competitive, but still leave larger deleted-interaction residuals despite relying on historical data or partitioned structures. Replay-based and data-free methods are more fragile: ER+GA trails on retained utility, while Naive+GA and EWC-style methods exhibit substantially larger drift or incomplete deletion. The full-user deletion results on MovieLens-20M in Table 4 show a similar trend under a stronger request type. ALIGN-REC obtains the smallest non-oracle Unlearning Gap, reducing it from the best competing value of 0.093 to 0.085, while also achieving the best retained-user HR@5, NDCG@5, and BWT. This indicates that removing a user's complete interaction history does not require sacrificing retained-user recommendation quality. In contrast, data-dependent methods remain strong but still retain more deleted-user signal, while replay-based and data-free baselines suffer a sharper instability–forgetting trade-off. Overall, these results demonstrate that ALIGN-REC handles heterogeneous user-centric unlearning requests across both targeted user–category deletion and complete user-history deletion while preserving retained recommendation behavior.

Table 4: Full-user unlearning with the **Two-Tower** backbone on **MovieLens-20M**. We report Unlearning Gap for deleted users and HR@5, NDCG@5, and BWT for retained users. Best result excluding *Oracle* is in **bold**; second best is underlined. $p < 0.05$ by a paired two-sided t-test vs the best competing method.

Metric	Oracle	Data-Dependent		Replay	Data-Free				%Δ
		RecEraser	UltraRE	ER+GA	Naive+GA	EWC+GA	EWC+SCIF	ALIGN-REC	
UGap ≈ 0	0.065	0.102	0.093	0.230	0.345	0.262	0.202	**0.085**	+8.60%
HR@5 ↑	0.867	0.767	0.793	0.735	0.685	0.743	0.758	**0.812**	+2.40%
NDCG@5 ↑	0.646	0.589	0.594	0.584	0.523	0.561	0.582	**0.600**	+1.01%
BWT ↑	0.004	-0.023	-0.018	-0.056	-0.212	-0.102	-0.082	**-0.013**	+27.78%

6.4 Robustness

We next stress test ALIGN-REC'S reliability under two stream-level modes (1) *long-horizon accumulation*, where increasing the number of episodes gives retained drift and deleted-signal regrowth more opportunities to compound. and (2) *high unlearning pressure*, where a larger fraction of unlearning requests forces the model to erase more frequently while preserving retained preferences. We summarize performance using *average rank* across HR@5, NDCG@5, BWT, and $|\mathrm{UGap}|$. Figure 5 shows that ALIGN-REC remains consistently best-ranked under both stress modes. On Amazon Reviews, as the number of episodes increases, ALIGN-REC maintains the lowest average rank, indicating that its retain/forget geometry limits its accumulated interference over longer horizons. On MovieLens-20M, as the fraction of unlearning requests increases from 15% to 50%, ALIGN-REC again remains the most stable method, showing robustness under heavier deletion workloads. The baselines degrade more noticeably under these stresses. Data-dependent methods such as RecEraser and UltraRE remain competitive but trail ALIGN-REC under repeated interleaving. ER+GA becomes sensitive to stream composition as replay buffers are repeatedly affected by deletions. Data-free methods such as Naive+GA, EWC+GA, and EWC+SCIF avoid historical storage, but show weaker robustness because post-hoc deletion or parameter-importance constraints alone do not sufficiently control retained drift and deleted-signal regrowth. Additional normalized radar visualizations are provided in Section H.2.



(a) Variable streams on Amazon Reviews. (b) Variable unlearn ratio on MovieLens.
Figure 5: Average rank (lower is better) across metrics over five seeds.

7 Conclusion

This paper investigates the problem of recommendation under interleaved Learn-Unlearn streams, where models must continuously adapt while selectively forgetting user-requested data without

compromising retained utility. To address this, we propose ALIGN-REC, a principled model-agnostic framework that resolves the problem of collaborative signal regrowth by regulating update geometry, thereby enabling stable continual updates and targeted unlearning. Extensive experiments on datasets validate that it achieves a strong performance across retained utility, stability, and forgetting quality under diverse conditions. We leave the question of improving the tightness of curvature surrogate and adaptive summary to further reduce overhead for future work (further discussion in Section J). Overall, ALIGN-REC provides a practical foundation for advancing continual heterogeneous unlearning in real-world settings.

Acknowledgments and Disclosure of Funding

We declare no additional competing interests beyond our disclosed affiliations.

References

[1] C. C. Aggarwal. *Recommender Systems: The Textbook*. 1st. Springer Publishing Company, Incorporated, 2016.

[2] K. Ahrabian et al. "Structure Aware Experience Replay for Incremental Learning in Graph-based Recommender Systems". In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. CIKM '21. Virtual Event, Queensland, Australia: Association for Computing Machinery, 2021, pp. 2832–2836. ISBN: 9781450384469.

[3] L. Bourtoule et al. "Machine unlearning". In: *2021 IEEE symposium on security and privacy (SP)*. IEEE. 2021, pp. 141–159.

[4] C. Chen et al. "Recommendation Unlearning". In: *Proceedings of the ACM Web Conference 2022*. WWW '22. Virtual Event, Lyon, France: Association for Computing Machinery, 2022.

[5] G. Chen et al. "Pre-training for Recommendation Unlearning". In: *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '25. Padua, Italy: Association for Computing Machinery, 2025, pp. 2112–2121. ISBN: 9798400715921.

[6] M. B. Cohen et al. "Input sparsity time low-rank approximation via ridge leverage score sampling". In: SODA '17. Barcelona, Spain: Society for Industrial and Applied Mathematics, 2017, pp. 1758–1777.

[7] R. D. Cook et al. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[8] P. Covington et al. "Deep Neural Networks for YouTube Recommendations". In: RecSys '16. New York, NY, USA: Association for Computing Machinery, 2016. ISBN: 9781450340359.

[9] R. A. Fisher. "Theory of statistical estimation". In: *Mathematical proceedings of the Cambridge philosophical society*. Vol. 22. 5. Cambridge University Press. 1925, pp. 700–725.

[10] A. Golatkar et al. " Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks ". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2020, pp. 9301–9309.

[11] C. Guo et al. "Certified data removal from machine learning models". In: *Proceedings of the 37th International Conference on Machine Learning*. ICML'20. JMLR.org, 2020.

[12] N. Halko et al. "Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions". In: *SIAM Review* 53 (May 2011), pp. 217–288.

[13] F. M. Harper et al. "The MovieLens Datasets: History and Context". In: *ACM Trans. Interact. Intell. Syst.* 5.4 (Dec. 2015). ISSN: 2160-6455.

[14] X. He et al. "Lightgcn: Simplifying and powering graph convolution network for recommendation". In: *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2020, pp. 639–648.

[15] X. He et al. "Neural Collaborative Filtering". In: *Proceedings of the 26th International Conference on World Wide Web*. WWW '17. Perth, Australia: International World Wide Web Conferences Steering Committee, 2017, pp. 173–182. ISBN: 9781450349130.

[16] W.-C. Kang et al. "Self-Attentive Sequential Recommendation". In: *2018 IEEE International Conference on Data Mining (ICDM)* (2018), pp. 197–206.

[17] J. Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[18] P. W. Koh et al. "Understanding black-box predictions via influence functions". In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17. Sydney, NSW, Australia: JMLR.org, 2017, pp. 1885–1894.

[19] Y. Li et al. "Making recommender systems forget: Learning and unlearning for erasable recommendation". In: *Know.-Based Syst.* 283.C (Jan. 2024). ISSN: 0950-7051.

[20] Y. Li et al. "Selective and collaborative influence function for efficient recommendation unlearning". In: *Expert Syst. Appl.* 234.C (Dec. 2023). ISSN: 0957-4174.

[21] Y. Li et al. "Ultrare: Enhancing receraser for recommendation unlearning via error decomposition". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 12611–12625.

[22] D. Lopez-Paz et al. "Gradient episodic memory for continual learning". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6470–6479. ISBN: 9781510860964.

[23] J. J. McAuley et al. "Image-based Recommendations on Styles and Substitutes". In: *CoRR* abs/1506.04757 (2015). arXiv: 1506.04757.

[24] F. Mi et al. "ADER: Adaptively Distilled Exemplar Replay Towards Continual Learning for Session-based Recommendation". In: *Proceedings of the 14th ACM Conference on Recommender Systems*. RecSys '20. Virtual Event, Brazil: Association for Computing Machinery, 2020. ISBN: 9781450375832.

[25] C. Musco et al. *Randomized Block Krylov Methods for Stronger and Faster Approximate Singular Value Decomposition*. 2015. arXiv: 1504.05477 [cs.DS].

[26] S.-A. Rebuffi et al. "icarl: Incremental classifier and representation learning". In: *2017 IEEE conference on computer vision and pattern recognition (CVPR)*. IEEE. 2017, pp. 5533–5542.

[27] F. Ricci et al. "Recommender systems: Techniques, applications, and challenges". In: *Recommender systems handbook* (2021), pp. 1–35.

[28] D. Rolnick et al. "Experience replay for continual learning". In: *Advances in neural information processing systems* 32 (2019).

[29] Y. Wang et al. "Structure aware incremental learning with personalized imitation weights for recommender systems". In: *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*. AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. ISBN: 978-1-57735-880-0.

[30] Y. Xu et al. "GraphSAIL: Graph Structure Aware Incremental Learning for Recommender Systems". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. CIKM '20. Virtual Event, Ireland: Association for Computing Machinery, 2020, pp. 2861–2868. ISBN: 9781450368599.

[31] X. Yi et al. "Sampling-bias-corrected neural modeling for large corpus item recommendations". In: *Proceedings of the 13th ACM conference on recommender systems*. 2019, pp. 269–277.

[32] X. Zhang et al. "Influential exemplar replay for incremental learning in recommender systems". In: *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*. AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN: 978-1-57735-887-9.

[33] J. Zhu et al. "Reloop2: Building self-adaptive recommendation models via responsive error compensation loop". In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023, pp. 5728–5738.

Index

- Related Works ..... p. 12
- Notation Table ..... p. 26
- Metrics ..... p. 12
- Algorithms ..... p. 13
- Time and Space Complexity ..... p. 13
- Implementation Details ..... p. 15
- Detailed Experimentation Details ..... p. 15

• Additional Results ..... p. 17
• Proofs ..... p. 19
• Discussion ..... p. 25

A Related Works

Continual learning (CL) helps to incorporate new data without catastrophic forgetting, commonly using regularization/importance penalties (e.g., EWC) or constrained updates (e.g., GEM), and rehearsal/distillation with stored exemplars [17, 22, 26]. In recommender systems, CL has been adapted to sequential and graph settings through structure-aware regularization that preserves historical dependencies under evolving graphs [30, 29], rehearsal/distillation strategies that replay representative samples or features during incremental updates [32, 24], and self-correction mechanisms that revise representations using error memory [33]. These CL methods primarily target retention and do not model explicit deletion-driven unlearning. In contrast, machine unlearning aims to remove the influence of specific training data while preserving utility [3, 10, 11], spanning shard/partition retraining (e.g., SISA) [3], certified removal under restricted assumptions [11], and influence-function approximations [18]. For recommender systems, prior work includes shard/ensemble-based approaches such as RecEraser, LASER, and UltraRE [4, 19, 21], influence-style correction methods such as SCIF [20], and pretraining-based strategies to improve unlearning readiness [5]. However, most unlearning work assumes offline or single-shot deletions and does not address interleaved Learn–Unlearn streams, where deleted signal can regrow under continued learning.

B Notation Table

Here we summarize all the notations used in the paper.

C Metrics

We evaluate next-item ranking using fixed candidate sets: for each test interaction, we rank the ground-truth item against M sampled negatives (default M=50), yielding $(M+1)$ candidates. We report retained-set utility on non-deleted interactions and forgetting quality on deleted interactions. We report metrics after the final request in the stream and average over random seeds.

Ranking utility (retained set). Let r be the rank position (1 = best) of the ground-truth item among $(M+1)$ candidates. We report:

$$\mathrm{HR@5} = \mathbb{I}[r \le 5], \tag{15}$$

$$\mathrm{NDCG@5} = \begin{cases} \frac{1}{\log_2(r+1)}, & r \le 5, \\ 0, & r > 5. \end{cases} \tag{16}$$

Stability (BWT). We evaluate after each request to track stability over the stream. Let $A_{t,j}$ denote HR@5 (or NDCG@5) on the retained test set for phase/task j evaluated after request t (with $j \le t$). Backward Transfer is:

$$\mathrm{BWT} = \frac{1}{T-1} \sum_{j=1}^{T-1} \big(A_{T,j} - A_{j,j}\big). \tag{17}$$

Values closer to 0 indicate minimal interference on earlier retained phases.

Unlearning Gap (deleted set). Let U be HR@5 or NDCG@5 computed on the deleted test set after unlearning. Let U^{rand} be the same metric obtained from a randomly initialized model over $(M+1)$ candidates. We define:

$$\mathrm{UGap} = U - U^{\text{rand}}. \tag{18}$$

Smaller $|\mathrm{UGap}|$ is better; $\mathrm{UGap} \approx 0$ indicates chance-level behavior on deleted interactions.

Membership Inference Attack (MIA AUC). We report ROC-AUC of a membership inference attack classifier trained to distinguish retained training interactions from held-out interactions using model outputs/statistics (e.g., score, loss, margin). An AUC near 0.5 indicates chance-level leakage; when comparing methods we treat smaller $|\mathrm{AUC} - 0.5|$ as better.

Average rank (robustness plots). For robustness summaries, we compute an average-rank score (lower is better). For each setting (e.g., number of episodes or unlearning percentage), we rank methods per metric using HR@5, NDCG@5, BWT, and $|\mathrm{UGap}|$, then average ranks across metrics and seeds.

D Algorithms

Algorithm 1 CurvatureSketch(G, r, q)

Require: : Gradients $G \in \mathbb{R}^{p \times N_f}$ (columns g_i), target rank r, power iters q

Draw $\Omega \sim \mathcal{N}(0,1)^{N_f \times r}$; set $Y \leftarrow G\Omega$
for $j \leftarrow 1$ **to** q **do**
    $Y \leftarrow G(G^\top Y)$
    $Y \leftarrow \mathrm{qr}(Y)$
end for
$\tilde{U} \leftarrow Y$; $\tilde{P} \leftarrow \tilde{U}\tilde{U}^\top$
$B \leftarrow \tilde{U}^\top \big(\frac{1}{N_f} GG^\top\big)\tilde{U} \in \mathbb{R}^{r \times r}$
$(Q, \Lambda) \leftarrow \mathrm{eig}(B)$
$\tilde{C} \leftarrow (\tilde{U}Q)\,\Lambda\,(\tilde{U}Q)^\top$
return $(\tilde{P}, \tilde{C})$

Algorithm 2 ForgetSummary$(G, \tilde{P}_f^t, \hat{C}, k_f, \tau)$

Require: Gradients $G = [g_1, \ldots, g_{N_f}]$, projector $\tilde{P}_f^t$, surrogate $\hat{C}$, sample size k_f, tolerance τ

Compute scores $s_i \leftarrow \|\tilde{P}_f^t g_i\|_2^2$; set $p_i \leftarrow s_i / \sum_{j=1}^{N_f} s_j$
Sample a multiset $\mathcal{I}_f$ of k_f indices i.i.d. from $\{1, \ldots, N_f\}$ with probabilities p_i
Define $\widehat{H} \leftarrow \frac{1}{k_f} \sum_{i \in \mathcal{I}_f} \frac{1}{N_f\, p_i} (\tilde{P}_f^t g_i)(\tilde{P}_f^t g_i)^\top$
Define $H \leftarrow \tilde{P}_f^t\, \hat{C}\, \tilde{P}_f^t$
Compute residual $e \leftarrow \|H - \widehat{H}\|_2$
while $e > \tau$ **do**
    Select $i^\star$ to reduce the residual
    $\mathcal{I}_f \leftarrow \mathcal{I}_f \cup \{i^\star\}$; update $\widehat{H}$; recompute e
end while
Aggregate $\mathcal{I}_f$ into a set S_f with weights $w = \{w_i\}_{i \in S_f}$ by setting
    $w_i \leftarrow \frac{1}{N_f} \cdot \frac{c_i}{|\mathcal{I}_f|} \cdot \frac{1}{p_i}$, where c_i is the count of index i in $\mathcal{I}_f$
return (S_f, w)

E Time and Space Complexity

Here, we summarize the per-step complexity of CARE and report empirical update-time measurements compared to the other baselines (Table 6). Let p be the number of trainable parameters, N_t the current request-batch size, N_f and N_r the forget and retain batch sizes, r the target subspace rank, q the number of randomized power iterations, and (k_f, k_r) the forget and retain summary budgets. CARE avoids explicitly forming or decomposing a $p \times p$ curvature matrix. The randomized curvature sketch costs $\mathcal{O}(q\, pN_t r)$ time and stores a rank-r basis using $\mathcal{O}(pr)$ memory. Summary construction uses projected gradients in this rank-r space. Forget-summary selection costs $\mathcal{O}(prN_f + k_f r^2)$, where the first term computes projected scores and the second constructs the compact weighted curvature estimate. Retain-summary selection costs $\mathcal{O}(prN_r + k_r N_r r)$ up to small lower-order

Algorithm 3 RetainSummary$(G_r, \tilde{P}_r^t, \hat{g}_r, W_r, k_r, \eta_r)$

Require: Gradients $G_r = [g_1, \ldots, g_{N_r}]$, projector $\tilde{P}_r^t$, running mean $\hat{g}_r$, normalization operator W_r, budget k_r, tolerance η_r

Initialize $S_r \leftarrow \varnothing$, residual $e_0 \leftarrow \hat{g}_r$
while $\|W_r e_0\|_2 > \eta_r$ **and** $|S_r| < k_r$ **do**
  $i^\star \leftarrow \arg\max_i \langle W_r \tilde{P}_r^t g_i,\, W_r e_0 \rangle$
  $S_r \leftarrow S_r \cup \{i^\star\}$
  Solve $w \in \Delta^{|S_r|}$ for
    $\min_w \ \big\| W_r \big(\sum_{i \in S_r} w_i \, \tilde{P}_r^t g_i - \hat{g}_r \big) \big\|_2$
  $e_0 \leftarrow \hat{g}_r - \sum_{i \in S_r} w_i \, \tilde{P}^{\perp} g_i$
end while
return (S_r, w)

Algorithm 4 CARE (Curvature-Aligned Retain–Erase)

Require: θ_{t-1}, step size η, inner steps S, rank r, power iters q, summary budgets (k_f, k_r), tolerances (τ, η_r), weights (λ_r, λ_f), margin m, EMA rates (μ, ν, ρ), stabilization ϵ, covariance statistic $\hat{C}$, EMA of projected retain gradients $\hat{g}_{r,t-1}$, diagonal curvature statistic $m_{r,t-1}$, normalization operator $W_{r,t-1}$

$\theta \leftarrow \theta_{t-1}$; initialize (S_t, w_t).
Receive request $R_t = (\sigma_t, \mathcal{D}_t)$.
Compute per-sample gradients $\{g_i\}_{i \in \mathcal{D}_t}$ at θ_{t-1} to form sketch/summaries; optimize θ for S steps using the resulting sets.
$G^t \leftarrow [g_i]_{i \in \mathcal{D}_t}$
$(\tilde{P}^t, \tilde{C}) \leftarrow \textsc{CurvatureSketch}(G^t, r, q)$
$\hat{C} \leftarrow (1-\rho)\hat{C} + \rho \, \tilde{C}$
if $\sigma_t = learn$ **then**
  $(S_t, w_t) \leftarrow \textsc{RetainSummary}(G, \tilde{P}^t, \hat{g}_r, W_{r,t-1}, k_r, \eta_r)$
else ▷ $\sigma_t = unlearn$
  $(S_t, w_t) \leftarrow \textsc{ForgetSummary}(G, \tilde{P}^t, \hat{C}, k_f, \tau)$
end if
for $s = 1, 2, \ldots, S$ **do**
  Compute $\mathcal{L}_{\text{task}}(\theta; S^t)$
  $F_{r,t-1} \leftarrow \operatorname{diag}(m_{r,t-1} + \epsilon)$
  $\mathcal{L}_{\text{stability}} \leftarrow \frac{\lambda_r}{2} \|\theta - \theta_{t-1}\|_{F_r}^2$
  if $\sigma_t = unlearn$ **then**
    $\mathcal{L}_{\text{erasure}} \leftarrow \frac{\lambda_f}{2} \|P(\theta - \theta_{t-1})\|_2^2$
  else
    $\mathcal{L}_{\text{erasure}} \leftarrow 0$
  end if
  $\theta \leftarrow \theta - \eta \, \nabla_\theta \Big(\mathcal{L}_{\text{task}} + \mathcal{L}_{\text{stability}} + \mathcal{L}_{\text{erasure}} \Big)$
end for
if $\sigma_t = learn$ **then**
  $\hat{g}_r \leftarrow (1-\mu)\hat{g}_r + \mu \, \frac{1}{|\mathcal{D}_t|} \sum_{i \in \mathcal{D}_t} \tilde{P}^t g_i$
  $m_r \leftarrow (1-\nu) m_r + \nu \operatorname{diag}\Big(\frac{1}{|\mathcal{D}_t|} \sum_{i \in \mathcal{D}_t} (\tilde{P}^t g_i) \odot (\tilde{P}^t g_i) \Big)$
  $W_r \leftarrow \operatorname{diag}\big((m_r + \epsilon)^{-1/2} \big)$
end if
return $\theta_t \leftarrow \theta$

terms from the summary-weight solver. Thus, the dominant additional CARE cost is linear in the parameter dimension and request-batch size, and depends on the low rank r rather than requiring quadratic $\mathcal{O}(p^2)$ curvature operations. The persistent state is $\mathcal{O}(pr + p + k_f + k_r)$ for the low-rank basis, retain-side EMA statistics, and summary indices/weights. In implementation, per-sample gradients can be processed in clusters, avoiding storage of full per-sample gradient histories.

As shown in Table 6, ALIGN-REC introduces moderate overhead relative to simpler gradient-ascent baselines, but remains within the range of competitive online methods and is faster than EWC+SCIF. UltraRec and RecEraser are faster per batch, but rely on stored shards or historical retained data making them unfeasible. ALIGN-REC instead avoids full-history storage through low-rank curvature sketches and compact retain/forget summaries, giving a practical trade-off between efficiency, retained utility, and unlearning quality.

Table 6: Average per-batch update time during the unlearning phase. Times are reported in milliseconds per batch under the same backbone, batch size, and hardware setting.

Metric	ULTRAREC	RecEraser	Naive+GA	EWC+GA	ER+GA	ALIGN-REC	EWC+SCIF
Time (ms/batch) ↓	26.12	27.34	34.12	35.45	36.21	40.08	45.73

F Implementation Details

In this section, we describe the implementation details of theexperiments. All experiments are conducted on a single NVIDIA A40 GPU. All methods are implemented in PyTorch and optimized with Adam. We use the same data splits, request streams, and evaluation checkpoints for all methods within each dataset/backbone setting to ensure fair comparison. We report mean over 5 random seeds. We evaluate our framework using a Two-Tower [8], NeuMF [15], LightGCN [14] and SASrec [16] recommender backbones. Training uses a fixed candidate construction with $M{=}50$ negatives per test instance (Appendix C). Within each dataset, we reuse the same optimizer settings (learning rate, batch size, and early stopping) across all compared methods; method-specific hyperparameters are tuned on a held-out validation stream and then fixed for reporting. Our continual setting is defined by an interleaved request stream that alternates learn and unlearn operations using the template $\langle L, L, U, L, U, L\rangle$. Amazon Reviews streams are constructed using item categories as tasks, whereas MovieLens-20M streams are constructed using release-year eras as tasks. After each request we evaluate for stability tracking; unless stated otherwise, tables report metrics after the final request. We group baselines into data-dependent, replay-based, and data-free families. Each baseline follows its canonical configuration and is adapted to the interleaved stream by applying the baseline's standard update on learn requests and its corresponding deletion step (if available) on unlearn requests. For replay-based methods, we remove interactions targeted by unlearning from the replay buffer; data-dependent methods (e.g., retraining and sharding-based approaches) access historical retained interactions as required by the baseline. ALIGN-REC is instantiated with CARE (Algorithm 4). On a learn request, CARE updates retain-side running statistics, constructs a retain summary under budget k_r, and optimizes the standard recommendation loss with a stability regularizer. On a unlearn request, CARE estimates a rank-r forget-subspace curvature surrogate via randomized subspace iteration (with q power iterations), constructs a forget summary under budget k_f, and optimizes a margin-based demotion loss together with an forget subspace suppression regularizer that suppresses update energy in the estimated forget subspace. We tune key CARE hyperparameters (including (k_f, k_r), r, λ_r, and λ_f) on the validation stream and select values from the stable regions identified by the sensitivity analysis (Figure 7). The final hyperparameter values used in experiments are reported in Table 7.

G Detailed Experimentation Details

This section provides additional details on datasets, preprocessing, request-stream construction, backbone models, and baselines. We report the mean over 5 random seeds and use identical data splits, request streams, candidate sets, and evaluation checkpoints across all methods within each dataset/backbone setting for fair comparison.

G.1 Datasets and preprocessing

We report main results on two public recommendation benchmarks:

- **Amazon Reviews** McAuley et al. [23]: implicit-feedback interactions derived from user reviews; we define stream slices/tasks by item categories.
- **MovieLens-20M** Harper et al. [13]: movie-rating interactions; we define stream slices/tasks by release-year eras to reflect temporal non-stationarity.

We time-order interactions per user, map user/item IDs to contiguous indices, and apply standard preprocessing used in recommender evaluation (e.g., removing extremely sparse users/items and invalid entries). All compared methods share the same processed data.

G.2 Candidate construction and evaluation

We evaluate next-item ranking using fixed candidate sets. For each test interaction, we rank the ground-truth item against M sampled negatives (default $M{=}50$), yielding $(M{+}1)$ candidates. HR@5 and NDCG@5 are computed on the retained (non-deleted) test set, while Unlearning Gap and MIA AUC are computed with respect to deleted interactions, following Appendix C.

G.3 Request streams and interleaving protocol

Our continual setting is defined by an interleaved request stream that alternates learn and unlearn operations using the fixed template $\langle L, L, U, L, U, L\rangle$, repeated over sequential slices. Each learn request updates the model on the next incoming slice, and each unlearn request deletes a previously observed slice according to the experiment setting. In robustness experiments, we vary either the number of episodes or the fraction of unlearning requests to control long-horizon accumulation and unlearning pressure.

- **Amazon Reviews:** slices correspond to item categories; unlearn requests delete interactions associated with the requested category slice.
- **MovieLens-20M:** slices correspond to release-year eras; unlearn requests delete interactions associated with the requested era slice.

Unless stated otherwise, tables report metrics after the final request in the stream. We evaluate after each request when computing stability (BWT) and robustness summaries (average rank).

G.4 Backbone model

To demonstrate the framework's efficacy, all methods are evaluated using a **Two-Tower** recommender backbone [31]. The user tower encodes a user representation from interaction history (or a user embedding), the item tower encodes an item representation, and the relevance score is computed via a similarity function (dot product) between the two embeddings. This retrieval-style architecture is widely used due to scalability and stable representation learning. We also demonstrate that the framework is model-agnostic by changing the backbone to NeuMF, LightGCN [14], and SASRec [16], covering neural collaborative filtering, graph-based collaborative filtering, and sequential recommendation. Within each dataset, we reuse the same optimizer setup, learning-rate schedule, and early stopping criterion across all methods, and only tune method-specific hyperparameters on a held-out validation stream.

G.5 Baselines and adaptation to interleaved streams

Since prior work does not directly target heterogeneous interleaved Learn–Unlearn streams in recommendation, we compare against strong baselines from three families and adapt them consistently to our protocol. All methods share the same request stream, data splits, candidate sets, and evaluation checkpoints within each compatible backbone setting. Concretely, on learn requests each baseline applies its canonical training update, and on unlearn requests it applies its corresponding deletion mechanism under evaluation settings.

- **Data-dependent unlearning.** *Retrain* serves as an upper bound by retraining from scratch on the retained interaction set after each unlearn request (i.e., excluding all deleted interactions up to that point). *RecEraser* is a sharding-based unlearning method that partitions training data into shards and removes the influence of deleted shards via aggregation over the remaining shard models [4]. In our interleaved setting, both methods are applied after each unlearn request using the same backbone and stream, but they require access to historical retained data/shards, which may be infeasible under strict data-retention constraints. We further include UltraRE [21] and UnlearnRec [5] as recent recommendation-unlearning baselines. Since some methods are architecture-specific, they are evaluated under their compatible backbone settings.

- **Replay-based continual learning.** We implement experience replay (ER) as a continual-learning baseline by maintaining a fixed-size buffer of past retained interactions and mixing replay samples with each incoming learn batch [28]. To support deletions, whenever an interaction is targeted by a unlearn request, it is removed from the buffer. We additionally evaluate *ER+GA*, where an explicit gradient-ascent deletion step is applied during unlearn episodes (while still using replay for subsequent learn updates). This family can preserve retained utility through rehearsal but relies on storing and continuously sanitizing a memory buffer as deletions arrive.
- **Data-free unlearning and parameter-importance baselines.** *Naive+GA* applies a direct gradient-ascent update on the deleted interactions at unlearn requests, without storing any historical data. *EWC+GA* extends this by regularizing parameter changes using importance weights (Fisher-style) as in Elastic Weight Consolidation [17], combined with the same deletion update at unlearn steps. *EWC+SCIF* further replaces the naive deletion direction with an influence/importance-style correction (SCIF) [20] to better target the effect of deleted interactions. All data-free methods operate without replay buffers or historical shards; in our protocol they apply their canonical learn update on learn requests and their unlearning update on unlearn requests.

All baseline hyperparameters are tuned on the validation stream under the same interleaved protocol and then fixed for reporting.

G.6 User-centric deletion protocols

For user-centric deletion experiments, we randomly sample 10% users per run. We evaluate two granularities: (i) *user-conditioned category unlearning* on Amazon Reviews, where we select a category and delete only the sampled users' interactions in that category, and (ii) *full-user unlearning* on MovieLens-20M, where we delete the sampled users' complete interaction history.

G.7 CARE Hyperparameters

ALIGN-REC is instantiated with CARE. On a learn request, CARE updates retain-side running statistics, constructs a retain summary, and optimizes the standard recommendation loss with a stability regularizer. On an unlearn request, CARE estimates a rank-r forget-subspace curvature surrogate via randomized subspace iteration, constructs a forget summary, and optimizes a margin-based demotion loss together with a forget-subspace anti-regrowth regularizer. The summary parameters k_f and k_r are specified as ratios; the corresponding summary budgets are $\lceil k_f|\mathcal{D}_f^t|\rceil$ and $\lceil k_r|\mathcal{D}_r^t|\rceil$. We tune key CARE hyperparameters, including (k_f, k_r), r, λ_r, and λ_f, on the validation stream and select values from the stable regions identified by the sensitivity analysis in Section H.4. The final hyperparameter values for the main Two-Tower experiments are reported in Table 7.

H Additional Results

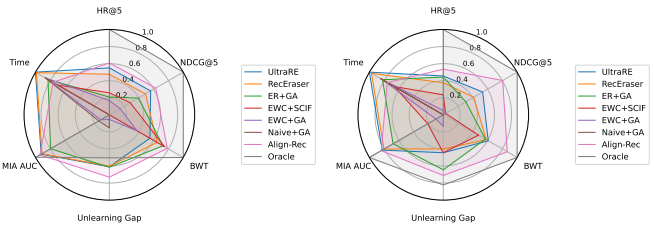
H.1 Learning-Unlearning Performance (Cont.) (RQ1)

We further validate the architecture generality of ALIGN-REC by instantiating it on NeuMF, LightGCN, and SASRec. These backbones cover neural collaborative filtering, graph-based collaborative filtering, and sequential recommendation, respectively. This evaluation complements the main Two-Tower results in Table 1 and tests whether the proposed geometry-aware update rule remains effective when deleted collaborative signal propagates through different model structures. As shown in Tables 9 to 11, ALIGN-REC consistently achieves the best or tied-best performance across retained utility, stability, and unlearning metrics. On NeuMF, ALIGN-REC improves HR@5 and NDCG@5 over the strongest competing method while reducing the absolute Unlearning Gap from 0.018 to 0.010 and moving MIA AUC closer to random guessing. On LightGCN, where deleted signal can spread through graph propagation, ALIGN-REC improves retained utility and BWT over UltraRE and UnlearnRec, while also obtaining the lowest absolute Unlearning Gap. On SASRec, ALIGN-REC again shows the strongest retained utility and stability, together with the smallest residual deleted-set behavior. Overall, the appendix results show trends consistent with the main paper: methods designed for one-shot or offline recommendation unlearning can be strong on isolated deletion steps, but they do not consistently preserve utility and suppress deleted signal under repeated interleaving. Continual-learning baselines preserve some adaptability but suffer from drift or incomplete deletion.

ALIGN-REC provides a more stable trade-off across all three additional architectures, supporting the claim that its retain/forget geometry is not tied to a specific recommender backbone.

H.2 Robustness (Cont.)(RQ3)

To complement the average-rank analysis in Section 6.4, we provide normalized radar visualizations on both MovieLens-20M and Amazon Reviews. Each metric is scaled to $[0, 1]$ so that higher values indicate better performance. The plots summarize retained utility (HR@5, NDCG@5), stability (BWT), unlearning quality (Unlearning Gap), inference attack behavior (MIA AUC), and time efficiency.



(a) MovieLens-20M. (b) Amazon Reviews.

Figure 6: **Normalized radar plots for additional robustness analysis.** Metrics are scaled to $[0, 1]$ with higher values indicating better performance. Across both datasets, ALIGN-REC shows the strongest overall trade-off among retained utility, stability, unlearning/privacy, and efficiency.

As shown in Figure 6, ALIGN-REC achieves the most balanced overall profile across both datasets. It remains closest to the oracle retraining reference on retained utility, stability, and forgetting-related metrics, while also maintaining competitive runtime. This indicates that the gains of ALIGN-REC are not limited to a single metric, but reflect a consistently strong trade-off across the main objectives of continual recommendation under unlearning requests.

Data-dependent baselines such as RecEraser can be competitive on some axes and sometimes faster, but they rely on stored shards or historical retained data, which weakens their applicability in strictly online settings. Replay-based and data-free baselines avoid this assumption, but they show weaker forgetting, stability, or retained utility. The radar plots therefore support the main robustness conclusion: ALIGN-REC provides the strongest overall utility–stability–unlearning trade-off while remaining practically efficient.

H.3 Ablation Study (RQ4)

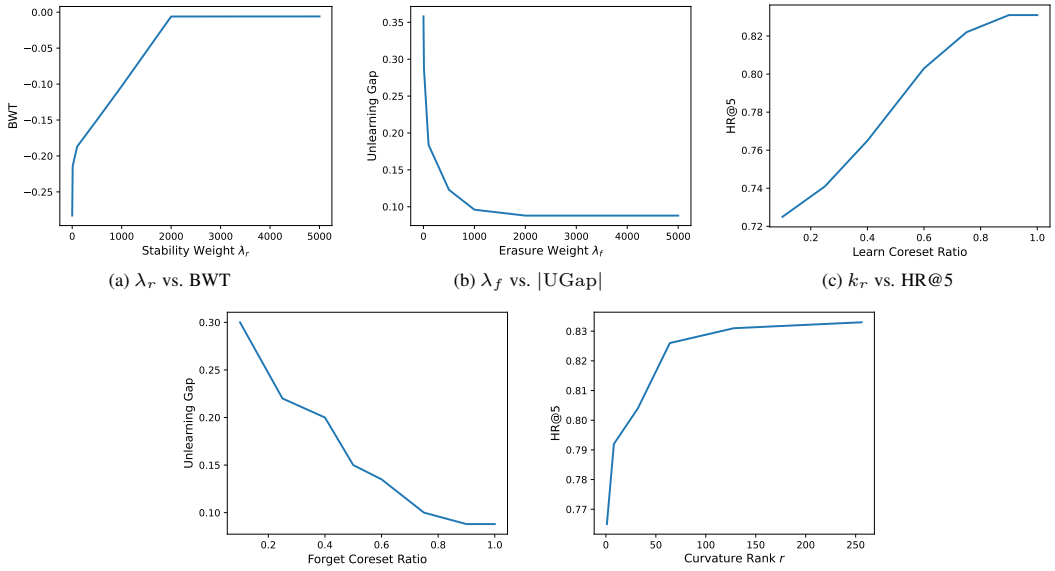
We ablate the main components of ALIGN-REC on MovieLens-20M with the Two-Tower backbone. Each variant removes one module while keeping the remaining training protocol fixed. This evaluates whether the geometry-aligned projector, retain/forget summaries, stability regularizer, and anti-regrowth regularizer are all necessary for the final utility–stability–unlearning trade-off.

Table 12: Ablation study of ALIGN-REC on **MovieLens-20M** with the **Two-Tower** backbone. We remove one component at a time. Best values are in **bold**. Results are averaged over 5 seeds; $p < 0.05$ by a paired two-sided t-test vs the best competing method.

Method	HR@5↑	NDCG@5↑	BWT↑	UGap≈ 0	MIA≈ 0.5
ALIGN-REC	**0.831**	0.632	**-0.006**	0.088	**0.511**
w/o geometry projector	0.792	0.593	-0.054	0.192	0.586
w/o retain summary	0.770	0.573	-0.128	0.113	0.570
w/o forget summary	0.819	0.615	-0.034	0.256	0.634
w/o stability reg.	0.724	0.535	-0.283	**0.081**	0.570
w/o anti-regrowth reg.	0.838	**0.639**	-0.020	0.358	0.678

Table 12 shows that each component contributes to a different part of the trade-off. Removing the geometry-aligned projector degrades all metrics, confirming that request-aligned update directions are important for controlling learn–unlearn interference. Removing the retain summary substantially hurts retained utility and BWT, showing that compact retained-gradient information is needed to preserve non-deleted preferences under continual updates. Removing the forget summary sharply increases both Unlearning Gap and MIA AUC, indicating that targeted forget supervision is necessary for reliable deletion.

The regularizers play complementary roles. Without the stability regularizer, retained utility and BWT collapse, showing that curvature-aware retention is needed to prevent drift. Without the anti-regrowth regularizer, retained utility slightly improves, but deletion quality deteriorates severely: Unlearning Gap increases from 0.088 to 0.358 and MIA AUC rises from 0.511 to 0.678. This supports the central claim that high retained utility alone does not imply successful unlearning; without explicit forget-subspace suppression, deleted signal can remain recoverable or regrow under future updates.



(a) λ_r vs. BWT (b) λ_f vs. $|\mathrm{UGap}|$ (c) k_r vs. HR@5

(d) k_f vs. $|\mathrm{UGap}|$ (e) r vs. HR@5

Figure 7: **Hyperparameter sensitivity of ALIGN-REC on MovieLens-20M.** We vary the retain-side stability weight λ_r, the forget-subspace anti-regrowth weight λ_f, the retain summary ratio k_r, the forget summary ratio k_f, and the curvature rank r. The results show stable saturation behavior: stronger regularization and larger summaries improve the relevant objective until the dominant retained or deleted directions are already captured.

Overall, the ablation validates that ALIGN-REC's gains come from combining geometry alignment, retained-signal anchoring, targeted forget summaries, and anti-regrowth regularization.

H.4 Hyperparameter Sensitivity (RQ5)

We analyze the sensitivity of ALIGN-REC to key hyperparameters on MovieLens-20M using the Two-Tower backbone. Figure 7 shows how each hyperparameter affects retained utility, stability, or unlearning quality. For unlearning, we report distance to the target behavior through $|\mathrm{UGap}|$, where lower is better. Overall, the trends are consistent with the intended roles of the components. Increasing the retain-side stability weight λ_r improves BWT and then saturates, showing that curvature-aware retention regularization reduces drift under interleaved updates. Increasing the forget-subspace anti-regrowth weight λ_f reduces $|\mathrm{UGap}|$ sharply before reaching a plateau, indicating that stronger suppression along deleted-signal directions improves unlearning until the dominant forget directions are already controlled. The summary budgets show similar saturation behavior. Increasing the retain summary ratio k_r improves HR@5, confirming that larger retained summaries better preserve non-deleted preferences. Increasing the forget summary ratio k_f reduces $|\mathrm{UGap}|$, showing that richer deleted-interaction summaries provide stronger supervision for removing targeted behavior. Finally, increasing the curvature rank r improves HR@5 rapidly at small ranks and then saturates, suggesting that a relatively low-rank curvature subspace is sufficient to capture the dominant geometry needed for stable updates. Larger ranks mainly increase computation with limited additional benefit.

I Proofs

I.1 Proof of Theorem 4.1

Proof. We begin by defining the tracking error $E_t \coloneqq C_t - \hat{C}_t$.

From Equation (2) and the update of $\hat{C}_t$, we have:

$$C_t = (1-\rho)C_{t-1} + \rho\,\bar{C}_t, \qquad \hat{C}_t = (1-\rho)\hat{C}_{t-1} + \rho\,\tilde{C}_t, \tag{i}$$

Now defining the tracking error using these values, we have:

$$\begin{aligned} E_t &= C_t - \hat{C}_t \\ &= \big((1-\rho)C_{t-1} + \rho\,\bar{C}_t\big) - \big((1-\rho)\hat{C}_{t-1} + \rho\,\tilde{C}_t\big) \\ &= (1-\rho)(C_{t-1} - \hat{C}_{t-1}) + \rho(\bar{C}_t - \tilde{C}_t) \\ &= (1-\rho)E_{t-1} + \rho(\bar{C}_t - \tilde{C}_t). \end{aligned} \tag{ii}$$

Iterating the recursion gives us:

$$\begin{aligned} E_t &= (1-\rho)\Big((1-\rho)E_{t-2} + \rho(\bar{C}_{t-1} - \tilde{C}_{t-1})\Big) + \rho(\bar{C}_t - \tilde{C}_t) \\ &= (1-\rho)^2 E_{t-2} + \rho(1-\rho)(\bar{C}_{t-1} - \tilde{C}_{t-1}) + \rho(\bar{C}_t - \tilde{C}_t) \\ &\vdots \\ &= (1-\rho)^t E_0 + \rho\sum_{\tau=1}^{t}(1-\rho)^{t-\tau}\,(\bar{C}_\tau - \tilde{C}_\tau). \end{aligned} \tag{iii}$$

Since $\hat{C}_0 = C_0$, we have $E_0 = C_0 - \hat{C}_0 = 0$, hence

$$C_t - \hat{C}_t = \rho\sum_{\tau=1}^{t}(1-\rho)^{t-\tau}\,(\bar{C}_\tau - \tilde{C}_\tau). \tag{iv}$$

Taking the spectral norms and applying the triangle inequality results in:

$$\|C_t - \hat{C}_t\|_2 = \left\|\rho\sum_{\tau=1}^{t}(1-\rho)^{t-\tau}\,(\bar{C}_\tau - \tilde{C}_\tau)\right\|_2 \quad \le \rho\sum_{\tau=1}^{t}(1-\rho)^{t-\tau}\,\|\bar{C}_\tau - \tilde{C}_\tau\|_2. \tag{v}$$

It remains to bound $\|\bar{C}_\tau - \tilde{C}_\tau\|_2$ for each $\tau \le t$. To achieve it, we fix $\tau \in \{1, \dots, t\}$ and set $A \coloneqq \bar{C}_\tau$.

Let the eigendecomposition of A be $A = U\Lambda U^\top$ with eigenvalues $\lambda_1(A) \ge \cdots \ge \lambda_p(A) \ge 0$, $U_r \in \mathbb{R}^{p\times r}$ be the top-r eigenvectors and define the optimal rank-r projector $P_\star \coloneqq U_r U_r^\top$.

Then the best rank-r approximation in spectral norm is $A_r \coloneqq P_\star A P_\star$, and

$$\|A - A_r\|_2 = \|A - P_\star A P_\star\|_2 = \lambda_{r+1}(A). \tag{vi}$$

Now, Algorithm 1 returns an orthonormal basis $\tilde{U} \in \mathbb{R}^{p\times r}$ and $\tilde{P} \coloneqq \tilde{U}\tilde{U}^\top$. It forms $B = \tilde{U}^\top A \tilde{U}$ and computes $(Q, \Lambda) = \mathrm{eig}(B)$, which gives us:

$$\tilde{C}_\tau = (\tilde{U}Q)\Lambda(\tilde{U}Q)^\top. \tag{vii}$$

Since $B = Q\Lambda Q^\top$, we have:

$$\tilde{C}_\tau = (\tilde{U}Q)(Q^\top B Q)(\tilde{U}Q)^\top \quad = \tilde{U}\,B\,\tilde{U}^\top = \tilde{U}(\tilde{U}^\top A\tilde{U})\tilde{U}^\top \quad = (\tilde{U}\tilde{U}^\top)A(\tilde{U}\tilde{U}^\top) = \tilde{P}\,A\,\tilde{P}. \tag{viii}$$

Therefore,

$$\begin{aligned} \|A - \tilde{C}_\tau\|_2 = \|A - \tilde{P}A\tilde{P}\|_2 \qquad\qquad = \\ \|(A - P_\star A P_\star) + (P_\star A P_\star - \tilde{P}A\tilde{P})\|_2 \le \|A - P_\star A P_\star\|_2 + \|P_\star A P_\star - \tilde{P}A\tilde{P}\|_2. \end{aligned} \tag{ix}$$

The first term is handled by Equation (vi):

$$\|A - P_\star A P_\star\|_2 = \lambda_{r+1}(A).$$

For the second term,

$$P_\star A P_\star - \tilde{P}A\tilde{P} = (P_\star - \tilde{P})AP_\star + \tilde{P}A(P_\star - \tilde{P}), \tag{x}$$

therefore using $\|P_\star\|_2 = \|\tilde{P}\|_2 = 1$ and submultiplicativity, we get:

$$\begin{aligned}\|P_\star A P_\star - \tilde{P} A \tilde{P}\|_2 &\le \|(P_\star - \tilde{P}) A P_\star\|_2 + \|\tilde{P} A (P_\star - \tilde{P})\|_2 \\ &\le \|P_\star - \tilde{P}\|_2 \, \|A\|_2 \, \|P_\star\|_2 + \|\tilde{P}\|_2 \, \|A\|_2 \, \|P_\star - \tilde{P}\|_2 \\ &= 2 \, \|A\|_2 \, \|P_\star - \tilde{P}\|_2 \\ &= 2 \, \lambda_1(A) \, \|P_\star - \tilde{P}\|_2.\end{aligned} \tag{xi}$$

For orthogonal projectors, we have $\|P_\star - \tilde{P}\|_2 = \|\sin\Theta(\mathcal{U}_r(A), \mathrm{span}(\tilde{U}))\|_2$. By the guarantee for randomized Block Krylov iteration (e.g., [25]), applied with a target failure probability of δ/t, we have that conditioned on $A = \bar{C}_\tau$, with probability at least $1 - \delta/t$:

$$\|P_\star - \tilde{P}\|_2 \le \frac{\lambda_{r+1}(A) + c_0 \, \lambda_1(A)/q^2}{\lambda_r(A) - \lambda_{r+1}(A)}, \tag{xii}$$

for a constant $c_0 > 0$.

By the theorem's well-separated dominant eigenspace assumption, the reciprocal eigengap factor $\lambda_1(A)/(\lambda_r(A) - \lambda_{r+1}(A))$ is bounded by an absolute constant (uniformly for $\tau \le t$), so substituting Equation (xii) into Equation (xi), on the same event, gives:

$$\|P_\star A P_\star - \tilde{P} A \tilde{P}\|_2 \le 2\lambda_1(A) \cdot \frac{\lambda_{r+1}(A) + c_0 \, \lambda_1(A)/q^2}{\lambda_r(A) - \lambda_{r+1}(A)} \quad \le c_1 \Big(\lambda_{r+1}(A) + \lambda_1(A)/q^2 \Big), \tag{xiii}$$

for an absolute constant $c_1 > 0$. Combining Equation (ix) with Equation (xiii) gives, on the same event,

$$\begin{aligned}\|A - \tilde{C}_\tau\|_2 &\le \lambda_{r+1}(A) + c_1 \Big(\lambda_{r+1}(A) + \lambda_1(A)/q^2 \Big) \\ &\le (1 + c_1) \lambda_{r+1}(A) + c_1 \lambda_1(A)/q^2 \\ &\le c \Big(\lambda_{r+1}(A) + \lambda_1(A)/q^2 \Big),\end{aligned} \tag{xiv}$$

where $c \coloneqq 1 + c_1$ absorbs the fixed constants. Recalling $A = \bar{C}_\tau$, gives us:

$$\|\bar{C}_\tau - \tilde{C}_\tau\|_2 \le c \Big(\lambda_{r+1}(\bar{C}_\tau) + \lambda_1(\bar{C}_\tau)/q^2 \Big). \tag{xv}$$

Applying a union bound over $\tau = 1, \ldots, t$ with per-step failure probability δ/t implies that, conditioned on $\{\bar{C}_\tau\}_{\tau=1}^t$, with probability at least $1 - \delta$, the bound above holds simultaneously for all $\tau \le t$. Substituting this into Equation (v) finally gives us:

$$\|C_t - \hat{C}_t\|_2 \le \rho \sum_{\tau=1}^{t} (1 - \rho)^{t-\tau} \, \Big(\lambda_{r+1}(\bar{C}_\tau) + c \, \lambda_1(\bar{C}_\tau)/q^2 \Big),$$

which is the desired result.

Hence Proved. □

I.2 Proof of Theorem 4.2

Proof. We fix a time step t and write $v_i \coloneqq \tilde{P}_f^t g_i \in \mathbb{R}^p$ for $i \in \{1, \ldots, N_f\}$. Recall from the main paper, that the forget-subspace curvature is given as:

$$H_f^t \;=\; \frac{1}{N_f} \sum_{i=1}^{N_f} v_i v_i^\top.$$

Also, Algorithm 2 draws i.i.d. indices $I_1, \ldots, I_{k_f}$ from $\{1, \ldots, N_f\}$ with $\mathbb{P}(I_\ell = i) = p_i$ and forms the weighted estimator

$$\widehat{H}_f^t \;=\; \frac{1}{k_f} \sum_{\ell=1}^{k_f} \frac{1}{N_f \, p_{I_\ell}} \, v_{I_\ell} v_{I_\ell}^\top.$$

We now define i.i.d. random matrices as:

$$X_\ell \;\coloneqq\; \frac{1}{N_f\, p_{I_\ell}}\, v_{I_\ell} v_{I_\ell}^\top, \qquad \ell = 1, \ldots, k_f, \tag{i}$$

such that $\widehat{H}_f^t = \frac{1}{k_f} \sum_{\ell=1}^{k_f} X_\ell$. Therefore, we have:

$$\mathbb{E}[X_\ell] = \sum_{i=1}^{N_f} p_i \cdot \frac{1}{N_f\, p_i}\, v_i v_i^\top = \frac{1}{N_f} \sum_{i=1}^{N_f} v_i v_i^\top = H_f^t, \tag{ii}$$

which gives us $\mathbb{E}[\widehat{H}_f^t] = H_f^t$.

Now, let $Y_\ell \coloneqq X_\ell - H_f^t$. Then $\mathbb{E}[Y_\ell] = 0$ and subsequently:

$$\widehat{H}_f^t - H_f^t = \frac{1}{k_f} \sum_{\ell=1}^{k_f} Y_\ell. \tag{iii}$$

We next bound $\|Y_\ell\|_2$ and the matrix variance term. Since $X_\ell \succeq 0$ and $H_f^t \succeq 0$, we can write:

$$\|Y_\ell\|_2 \le \|X_\ell\|_2 + \|H_f^t\|_2. \tag{iv}$$

Since, we know:

$$\|X_\ell\|_2 = \left\| \frac{1}{N_f\, p_{I_\ell}}\, v_{I_\ell} v_{I_\ell}^\top \right\|_2 = \frac{1}{N_f\, p_{I_\ell}}\, \|v_{I_\ell}\|_2^2 \le \max_i \frac{\|v_i\|_2^2}{N_f\, p_i} \le M, \tag{v}$$

where the last inequality is the theorem assumption. Therefore, we can write Equation (iv) as:

$$\|Y_\ell\|_2 \le M + \|H_f^t\|_2 =: R. \tag{vi}$$

Next, we define σ^2 as:

$$\sigma^2 \;\coloneqq\; \left\| \sum_{\ell=1}^{k_f} \mathbb{E}[Y_\ell^2] \right\|_2 = k \left\| \mathbb{E}[Y_1^2] \right\|_2. \tag{vii}$$

Using $(A - B)^2 \preceq 2A^2 + 2B^2$ for symmetric matrices, we have:

$$Y_1^2 = (X_1 - H_f^t)^2 \preceq 2X_1^2 + 2H_f^{t2},$$

hence

$$\left\| \mathbb{E}[Y_1^2] \right\|_2 \le 2 \left\| \mathbb{E}[X_1^2] \right\|_2 + 2\|H_f^t\|_2^2. \tag{viii}$$

We now compute X_1^2 explicitly. Since $(v_i v_i^\top)^2 = \|v_i\|_2^2\, v_i v_i^\top$, X_1^2 can be written as:

$$X_1^2 = \left(\frac{1}{N_f\, p_{I_1}} \right)^2 (v_{I_1} v_{I_1}^\top)^2 = \left(\frac{1}{N_f\, p_{I_1}} \right)^2 \|v_{I_1}\|_2^2\, v_{I_1} v_{I_1}^\top. \tag{ix}$$

Taking expectation gives us:

$$\mathbb{E}[X_1^2] = \sum_{i=1}^{N_f} p_i \left(\frac{1}{N_f\, p_i} \right)^2 \|v_i\|_2^2\, v_i v_i^\top = \frac{1}{N_f^2} \sum_{i=1}^{N_f} \frac{\|v_i\|_2^2}{p_i}\, v_i v_i^\top. \tag{x}$$

Using the normalized leverage bound $\frac{\|v_i\|_2^2}{N_f\, p_i} \le M$ (and equivalently $\frac{\|v_i\|_2^2}{p_i} \le N_f M$), we get:

$$\left\| \mathbb{E}[X_1^2] \right\|_2 \le \frac{1}{N_f^2} \left(\max_i \frac{\|v_i\|_2^2}{p_i} \right) \left\| \sum_{i=1}^{N_f} v_i v_i^\top \right\|_2 \le \frac{1}{N_f^2} (N_f M) \cdot \|N_f H_f^t\|_2 = M \|H_f^t\|_2. \tag{xi}$$

Therefore,

$$\left\| \mathbb{E}[Y_1^2] \right\|_2 \le 2M \|H_f^t\|_2 + 2\|H_f^t\|_2^2 \le c_0\, M \|H_f^t\|_2, \tag{xii}$$

for an absolute constant $c_0 > 0$ (absorbing the $\|H_f^t\|_2^2$ term into the constant). Consequently, we get:

$$\sigma^2 = k_f \left\| \mathbb{E}[Y_1^2] \right\|_2 \le c_0\, k_f\, M \|H_f^t\|_2. \tag{xiii}$$

Applying the matrix Bernstein inequality to the sum of the zero-mean random matrices Y_ℓ yields, for any threshold $\epsilon > 0$,

$$\Pr\left(\left\|\sum_{\ell=1}^{k_f} Y_\ell\right\|_2 \geq \epsilon\right) \leq 2r \cdot \exp\left(\frac{-\epsilon^2/2}{\sigma^2 + R\epsilon/3}\right), \tag{xiv}$$

where the prefactor uses the effective dimension r since all matrices Y_ℓ are supported on the rank-r forget subspace.

We now convert this to a bound on the average error $\|\widehat{H}_f^t - H_f^t\|_2 = \|\frac{1}{k_f}\sum Y_\ell\|_2$. Let ε denote the error threshold for the average. Setting $\epsilon = k\varepsilon$ in Equation (xiv), we obtain:

$$\Pr\Big(\|\widehat{H}_f^t - H_f^t\|_2 \geq \varepsilon\Big) = \Pr\left(\left\|\sum_{\ell=1}^{k_f} Y_\ell\right\|_2 \geq k_f\varepsilon\right) \leq 2r \cdot \exp\left(\frac{-(k_f\varepsilon)^2/2}{\sigma^2 + Rk_f\varepsilon/3}\right). \tag{xv}$$

To ensure this failure probability is at most δ, we require

$$2r \cdot \exp\left(\frac{-k_f^2\varepsilon^2/2}{\sigma^2 + Rk_f\varepsilon/3}\right) \leq \delta \quad \Longrightarrow \quad \frac{k_f^2\varepsilon^2/2}{\sigma^2 + Rk_f\varepsilon/3} \geq \log(2r) + \log(1/\delta).$$

Let $L \coloneqq \log r + \log(1/\delta)$ (absorbing constants into the log term). Rearranging for ε, we essentially need $k_f^2\varepsilon^2 \gtrsim 2L(\sigma^2 + Rk_f\varepsilon)$. For sufficiently large k_f, the variance term σ^2 dominates the range term $Rk_f\varepsilon$. Solving $k_f^2\varepsilon^2 \geq 2L\sigma^2$, gives the condition $\varepsilon \geq \frac{\sqrt{2L\sigma^2}}{k_f}$.

Finally, substituting the variance bound $\sigma^2 \leq c_0 k_f M \|H_f^t\|_2$ from Equation (xiii) gives

$$\varepsilon \geq \frac{\sqrt{2L(c_0 k_f M\|H_f^t\|_2)}}{k_f} = \sqrt{\frac{2c_0 M\|H_f^t\|_2\, L}{k_f}}. \tag{xvi}$$

Absorbing constants into a single absolute constant c, we get that with probability at least $1-\delta$:

$$\|\widehat{H}_f^t - H_f^t\|_2 \;\leq\; c\sqrt{\frac{M\,\|H_f^t\|_2\big(\log r + \log(1/\delta)\big)}{k_f}}.$$

Hence Proved. □

I.3 Proof of Corollary 4.3

Proof. Let $u \coloneqq \tilde{P}_f^t(\theta - \theta_{t-1}) \in \mathbb{R}^p$. Then the left-hand side of Equation (12) can be written as:

$$\left|\tfrac{\lambda_f}{2}\,\langle u,\, \widehat{H}_f^t u\rangle - \tfrac{\lambda_f}{2}\,\langle u,\, H_f^t u\rangle\right| = \tfrac{\lambda_f}{2}\,\big|\langle u,\, (\widehat{H}_f^t - H_f^t)\,u\rangle\big|. \tag{i}$$

Since $\widehat{H}_f^t - H_f^t$ is symmetric, we can bound its quadratic form by the spectral norm:

$$\big|\langle u,\, (\widehat{H}_f^t - H_f^t)\,u\rangle\big| \leq \|\widehat{H}_f^t - H_f^t\|_2\,\|u\|_2^2. \tag{ii}$$

Under the assumption $\|\widehat{H}_f^t - H_f^t\|_2 \leq \varepsilon$, we obtain

$$\tfrac{\lambda_f}{2}\,\big|\langle u,\, (\widehat{H}_f^t - H_f^t)\,u\rangle\big| \leq \tfrac{\lambda_f}{2}\,\varepsilon\,\|u\|_2^2 = \tfrac{\lambda_f}{2}\,\varepsilon\,\big\|\tilde{P}_f^t(\theta - \theta_{t-1})\big\|_2^2,$$

Hence Proved. □

I.4 Proof of Theorem 5.1

Proof. For a given step t, we define the per-step objective excluding the erasure term as:

$$J_t(\theta) \;\coloneqq\; \mathcal{L}_{\text{task}}(\theta; S^t) \;+\; \mathcal{L}_{\text{stability}}(\theta), \tag{i}$$

The objective minimized in Equation (13) is therefore:

$$J_t(\theta) \;+\; \mathbb{I}_{[\sigma_t = unlearn]} \cdot \frac{\lambda_f}{2} \big\| \tilde{P}_f^t(\theta - \theta_{t-1}) \big\|_2^2. \tag{ii}$$

Now, let $g_t \coloneqq \nabla J_t(\theta_{t-1})$. Assume J_t is L-smooth in a neighborhood of θ_{t-1}, i.e., $\|\nabla J_t(\theta) - \nabla J_t(\theta')\|_2 \le L\|\theta - \theta'\|_2$ locally. Then for any displacement $\Delta\theta$, we have the standard quadratic upper bound given as:

$$J_t(\theta_{t-1} + \Delta\theta) \;\le\; J_t(\theta_{t-1}) \;+\; \langle g_t, \Delta\theta \rangle \;+\; \frac{L}{2}\|\Delta\theta\|_2^2. \tag{iii}$$

Using the L-smoothness upper bound, we analyze the local quadratic surrogate of the per-step objective around θ_{t-1}. The resulting surrogate minimizer induces the update direction analyzed below.

Substituting the upper bound from Equation (iii) into the total objective gives the following quadratic surrogate:

$$\min_{\Delta\theta} \Big[\underbrace{J_t(\theta_{t-1}) + \langle g_t, \Delta\theta \rangle + \frac{L}{2}\|\Delta\theta\|_2^2}_{\text{Upper bound for } J_t(\theta_{t-1}+\Delta\theta)} \;+\; \underbrace{\mathbb{I}_{[\sigma_t = unlearn]} \cdot \frac{\lambda_f}{2} \|\tilde{P}_f^t \Delta\theta\|_2^2}_{\text{forget subspace suppression regularizer}} \Big]. \tag{iv}$$

Since $J_t(\theta_{t-1})$ is constant with respect to $\Delta\theta$, it does not affect the minimizer. Dropping it gives us the following quadratic optimization problem:

$$\min_{\Delta\theta \in \mathbb{R}^p} \Big\{ \langle g_t, \Delta\theta \rangle + \frac{L}{2}\|\Delta\theta\|_2^2 + \mathbb{I}_{[\sigma_t = unlearn]} \cdot \frac{\lambda_f}{2} \|\tilde{P}_f^t \Delta\theta\|_2^2 \Big\}. \tag{v}$$

The objective in Equation (v) is strictly convex, hence has a unique minimizer $\Delta\theta$. Its first-order optimality condition is given by:

$$g_t \;+\; L\,\Delta\theta \;+\; \mathbb{I}_{[\sigma_t = unlearn]} \cdot \lambda_f \, \tilde{P}_f^t \Delta\theta \;=\; 0, \tag{vi}$$

which is equivalently:

$$\big(LI + \mathbb{I}_{[\sigma_t = unlearn]} \cdot \lambda_f \tilde{P}_f^t\big)\Delta\theta \;=\; -g_t. \tag{vii}$$

Applying $\tilde{P}_f^t$ to both sides and using $(\tilde{P}_f^t)^2 = \tilde{P}_f^t$ gives us:

$$\big(L + \mathbb{I}_{[\sigma_t = unlearn]} \cdot \lambda_f\big)\tilde{P}_f^t \Delta\theta = -\tilde{P}_f^t g_t,$$

hence

$$\tilde{P}_f^t \Delta\theta = -\frac{1}{L + \mathbb{I}_{[\sigma_t = unlearn]} \cdot \lambda_f} \, \tilde{P}_f^t g_t. \tag{viii}$$

Now consider the corresponding update $\Delta\theta_0$ obtained from the same quadratic model with $\lambda_f = 0$. Its optimality condition is $g_t + L\,\Delta\theta_0 = 0$, i.e.,

$$\Delta\theta_0 = -\frac{1}{L}\, g_t \qquad \Longrightarrow \qquad \tilde{P}_f^t \Delta\theta_0 = -\frac{1}{L}\, \tilde{P}_f^t g_t. \tag{ix}$$

Combining Equation (viii) and Equation (ix) gives us:

$$\tilde{P}_f^t \Delta\theta = \frac{L}{L + \mathbb{I}_{[\sigma_t = unlearn]} \cdot \lambda_f} \, \tilde{P}_f^t \Delta\theta_0. \tag{x}$$

Taking squared norms results in:

$$\|\tilde{P}_f^t \Delta\theta\|_2^2 = \Big(\frac{L}{L + \mathbb{I}_{[\sigma_t = unlearn]} \cdot \lambda_f}\Big)^2 \, \|\tilde{P}_f^t \Delta\theta_0\|_2^2. \tag{xi}$$

In particular, when $\sigma_t = unlearn$, this becomes

$$\|\tilde{P}_f^t \Delta\theta\|_2^2 \;\le\; \Big(\frac{L}{L + \lambda_f}\Big)^2 \, \|\tilde{P}_f^t \Delta\theta_0\|_2^2,$$

which is the desired bound.

Hence Proved. □

J Discussion

ALIGN-REC reframes continual recommendation unlearning as a geometry-aware online update problem. Instead of treating learning and unlearning as separate operations, the framework regulates how each request moves the model in parameter space. This separation is useful in practice as the backbone recommender determines how user–item relevance is represented, while ALIGN-REC determines how incoming learn and unlearn requests should modify that representation without storing full raw interaction histories or retraining from scratch. Our experiments show that this design improves the retained-utility, unlearning, and stability trade-off across multiple deletion granularities, stream settings, and recommender backbones.

However, it is important to clarify the scope of the unlearning notion considered here. ALIGN-REC targets behavioral unlearning in ranking space. The deleted interactions should lose residual ranking influence, and this suppression should remain stable under later learning updates. The framework is therefore not focusing on **exact retraining-equivalent machine unlearning** for arbitrary future queries. This scope is natural for online recommendation, where the operational requirement is often to suppress specified user–item preferences while preserving useful behavior on retained interactions. The deleted set, evaluation protocol, and unlearning metric define the behavioral target. Having said that, richer deletion specifications or stronger metrics can be incorporated without changing the overall update framework.

The summary and subspace components are practical control interfaces. The forget summary specifies which deleted-gradient directions must be preserved for effective demotion, while the retain summary specifies which retained-gradient statistics should anchor future utility. Their quality depends on the chosen summary budgets, curvature rank, and request-batch gradients. In settings with more complex deletion policies or stronger privacy requirements, these budgets can be increased, the rank can be enlarged, or the summaries can be constructed from more conservative deletion neighborhoods. Conversely, when latency is more constrained, smaller summaries and lower-rank sketches provide a lighter-weight operating point at the cost of weaker geometric resolution.

From a deployment perspective, ALIGN-REC inherits the cost of online gradient-based recommendation updates. It avoids full-history storage and quadratic curvature computation, but it still requires per-request gradient statistics, low-rank sketching, and summary maintenance. This makes the method most appropriate for settings where retraining or shard maintenance is too expensive, but moderate online update overhead is acceptable. The empirical runtime analysis suggests that this overhead remains competitive with online continual/unlearning baselines, while providing a stronger utility–unlearning trade-off. In large-scale systems, the same design can be implemented with mini-batch gradient clustering, approximate sketches, or asynchronous summary updates to further reduce latency.

Finally, ALIGN-REC can be extended in several directions. The current experiments focus on representative public recommendation datasets, standard ranking metrics, and commonly used recommender backbones. Future work can study longer industrial streams, session-level and multi-modal recommenders. The framework can also incorporate improved curvature estimators, adaptive rank selection, or deletion-aware negative sampling. Overall, ALIGN-REC provides a practical foundation for continual recommendation under heterogeneous unlearning requests. As recommender backbones, deletion metrics become stronger, the same geometry-aware update principle can use them to make more reliable online learn–unlearn decisions.

Table 5: Summary of notations used in ALIGN-REC and CARE.

Symbol	Description
Stream setup and model	
t, T	Time step and total number of steps.
$\mathcal{D}$	Full interaction dataset; split into $\mathcal{D}^{\text{train}}$, $\mathcal{D}^{\text{val}}$, $\mathcal{D}^{\text{test}}$.
$R_t = (\sigma_t, \mathcal{D}_t)$	Request at step t.
σ_t	Request type, $\sigma_t \in \{\texttt{Learn}, \texttt{Unlearn}\}$.
$\mathcal{D}_t$	Interaction batch at step t, $\mathcal{D}_t = \mathcal{D}_t^r$ if $\sigma_t = \texttt{Learn}$, else $\mathcal{D}_t = \mathcal{D}_t^f$.
$\mathcal{D}_t^r$, $\mathcal{D}_t^f$	Retain batch / Forget batch at step t.
N_t, N_r, N_f	Batch sizes $\lvert\mathcal{D}_t\rvert$, $\lvert\mathcal{D}_t^r\rvert$, $\lvert\mathcal{D}_t^f\rvert$.
(x_i, y_i)	A user–item interaction pair; $x_i \in \mathcal{X} \subseteq \mathbb{R}^n$ features, $y_i \in \{0, 1\}$ label.
$f_{\theta_t} : \mathcal{X} \to \mathbb{R}$	Scoring function at step t.
$\theta_t \in \mathbb{R}^p$	Model parameters at step t.
$\Phi(\theta_{t-1}; R_t)$	Online update operator; $\theta_t = \Phi(\theta_{t-1}; R_t)$.
Geometry tracking (curvature / subspaces)	
g_i	Per-sample gradient at θ_{t-1}, e.g., $g_i = \nabla_\theta \ell(f_\theta(x_i), y_i)\lvert_{\theta_{t-1}}$.
$G_t = [g_1, \ldots, g_N]$	Per-sample gradient matrix for the current request batch.
$\bar{C}_t$	Instantaneous gradient covariance, $\bar{C}_t = \frac{1}{N} G_t G_t^\top$.
C_t	EMA covariance, $C_t = (1-\rho)C_{t-1} + \rho \bar{C}_t$.
ρ	EMA rate for covariance tracking.
r	Target subspace rank / sketch dimension ($r \ll p$).
$\tilde{U}_t \in \mathbb{R}^{p \times r}$	Estimated top-r curvature directions (basis).
$\Lambda_t \in \mathbb{R}^{r \times r}$	Diagonal spectrum associated with $\tilde{U}_t$.
$\hat{C}_t = \tilde{U}_t \Lambda_t \tilde{U}_t^\top$	Rank-r curvature surrogate.
$\tilde{P}_t = \tilde{U}_t \tilde{U}_t^\top$	Request-subspace projector.
$\tilde{P}_t^f$, $\tilde{P}_t^r$	Forget-subspace / Retain-subspace projector.
$H_t = \tilde{P}_t \hat{C}_t \tilde{P}_t$	Request-subspace curvature proxy.
H_t^f	Forget-subspace curvature (restriction of H_t to $\tilde{P}_t^f$).
Summaries	
S_t^f, S_t^r	Forget summary / Retain summary (coreset) at step t.
$w = \{w_i\}$	Coreset weights for selected indices.
k_f, k_r	Forget / Retain coreset budgets.
gap_t	Spectral gap/conditioning signal used to scale forget-summary tolerance (e.g., via $\lambda_r(\hat{C}_t)$).
α, $f(\cdot)$	Tolerance scale and gap-dependent (non-increasing) function in the forget-summary constraint.
$\hat{H}_t$	Weighted coreset estimate of forget curvature (built from S_t^f and w).
τ	Refinement tolerance used inside the forget-summary construction.
δ	Failure probability in probabilistic guarantees.
M	Upper bound constant used in sampling/approximation guarantees.
Retain-side running statistics	
$\hat{g}_{r,t}$	EMA of projected retain gradients.
$m_{r,t}$	EMA diagonal curvature statistic on retain subspace.
μ, ν	EMA rates for $\hat{g}_{r,t}$ and $m_{r,t}$.
ϵ	Diagonal stabilization constant.
$W_{r,t}$	Normalization operator, e.g., $\text{diag}(m_{r,t} + \epsilon)^{-1/2}$.
$F_{r,t}$	Retain metric, e.g., $\text{diag}(m_{r,t} + \epsilon)$.
Objectives / regularizers	
L_{task}	Task loss (retain-side base recommendation objective).
L_{forget}	Behavioral unlearning loss (margin-based demotion on S_t^f).
$L_{\text{stability}}$	Stability regularizer (penalizes drift in retain-sensitive directions).
L_{erasure}	forget subspace suppression regularizer (suppresses update energy in the forget subspace).
λ_r, λ_f	Weights for $L_{\text{stability}}$ and L_{erasure}.
m	Margin parameter in L_{forget}.
$\Pi(\cdot, \cdot)$	Pairing distribution for sampling retain/forget examples (when applicable).
Evaluation	
M_{neg}	# negatives per test interaction (default 50), giving $(M_{\text{neg}} + 1)$ candidates.
r_{pos}	Rank position of the ground-truth item among candidates (avoid confusion with subspace rank r).
HR@5, NDCG@5	Ranking utility at cutoff 5 on retained interactions.
BWT	Backward transfer (stability across retained phases).
UGap	Unlearning Gap on deleted interactions (target ≈ 0).
MIA AUC	Membership inference ROC-AUC (target ≈ 0.5).

Table 7: Selected hyperparameters for ALIGN-REC/CARE (tuned on validation).

Setting	k_f	k_r	r	λ_r	λ_f
Amazon / Two-Tower	0.75	0.75	64	1000.0	2000
MovieLens-20M / Two-Tower	0.8	0.8	128	2000	1000

Table 8: Statistics of the datasets.

Dataset	# Users	# Items	# Interactions	#Tasks
Amazon Reviews	154530	79902	1,650,320	10
MovieLens-20M	138493	27,278	20,000,263	10

Table 9: Learning–unlearning utility with the **NeuMF** backbone on **Amazon Reviews**. Best result excluding *Oracle* is in **bold**; second best is underlined.Results are averaged over 5 seeds; $p < 0.05$ by a paired two-sided t-test vs the best competing method.

Metric	**Oracle**	**Data-Dependent**		**Replay**	**Data-Free**				**%Δ**
		RecEraser	**UltraRE**	**ER+GA**	**Naive+GA**	**EWC+GA**	**EWC+SCIF**	**ALIGN-REC**	
HR@5 ↑	0.940	0.751	0.775	0.772	0.636	0.651	0.708	**0.801**	+3.35%
NDCG@5 ↑	0.820	0.613	0.655	0.585	0.476	0.487	0.538	**0.755**	+15.27%
BWT ↑	-0.012	-0.186	-0.175	-0.188	-0.416	-0.418	-0.226	**-0.068**	+61.14%
UGap ≈ 0	-0.005	0.025	0.023	0.018	-0.051	-0.044	-0.028	**-0.010**	+44.44%
MIA AUC ≈ 0.5	0.500	0.517	0.515	0.524	0.575	0.567	0.559	**0.511**	+26.67%

Table 10: Learning–unlearning utility with the **LightGCN** backbone on **Amazon Reviews**. Best result excluding *Oracle* is in **bold**; second best is underlined. Results are averaged over 5 seeds; $p < 0.05$ by a paired two-sided t-test vs the best competing method.

Metric	**Oracle**	**Data-Dependent**		**Pre-train**	**Replay**	**Data-Free**				**%Δ**
		RecEraser	**UltraRE**	**UnlearnRec**	**ER+GA**	**Naive+GA**	**EWC+GA**	**EWC+SCIF**	**ALIGN-REC**	
HR@5 ↑	0.960	0.756	0.780	0.765	0.765	0.630	0.645	0.702	**0.825**	+5.77%
NDCG@5 ↑	0.840	0.622	0.665	0.727	0.550	0.448	0.458	0.506	**0.780**	+7.29%
BWT ↑	-0.018	-0.196	-0.185	-0.271	-0.210	-0.465	-0.467	-0.252	**-0.075**	+59.46%
UGap ≈ 0	-0.002	0.038	+0.035	-0.133	+0.060	-0.169	-0.146	-0.094	**+0.025**	+28.57%
MIA AUC ≈ 0.5	0.500	0.523	0.520	0.540	0.542	0.631	0.618	0.603	**0.520**	+0.00%

Table 11: Learning–unlearning utility with the **SASRec** backbone on **Amazon Reviews**. Best result excluding *Oracle* is in **bold**; second best is underlined. Results are averaged over 5 seeds; $p < 0.05$ by a paired two-sided t-test vs the best competing method.

Metric	**Oracle**	**Data-Dependent**		**Replay**	**Data-Free**				**%Δ**
		RecEraser	**UltraRE**	**ER+GA**	**Naive+GA**	**EWC+GA**	**EWC+SCIF**	**ALIGN-REC**	
HR@5 ↑	0.975	0.775	0.800	0.785	0.647	0.662	0.720	**0.845**	+5.63%
NDCG@5 ↑	0.865	0.641	0.685	0.595	0.484	0.496	0.547	**0.805**	+17.52%
BWT ↑	-0.010	-0.180	-0.170	-0.180	-0.399	-0.401	-0.216	**-0.065**	+61.76%
UGap ≈ 0	-0.001	0.032	+0.030	+0.028	-0.079	-0.068	-0.044	**+0.018**	+35.71%
MIA AUC ≈ 0.5	0.500	0.521	0.518	0.531	0.597	0.587	0.576	**0.516**	+11.11%

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: The abstract and introduction accurately describe the contents of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of the work are discussed in the Discussion section in *Section J* in *Appendix*.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We have provided the assumptions and complete proofs in *Section 4 and Appendix I*.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have explained the framework of the proposed model in the main paper as well as the data preprocessing and other experiment details and implementation in *Section 3, Appendix F and G* in *Appendix*.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Our implementation will be available upon request. Implementation and experimental details are provided in Appendices F and G.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We state the training and test strategies at the beginning of the Experiments section, with the full details provided in *Appendix F* .

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in *Section D and E in supplementary* material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The experiments are conducted with statistical significance.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The computing resources are reported wherever necessary.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: Our work is statistical methodology/theory to ensure guaranteed performance in sequential recommendation models. To the best of our knowledge, there are no direct societal impacts of our work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: We credit the original datasets and baseline methods wherever necessary, but the paper does not document their licenses and terms of use.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.