SAF: SELECTIVE CONSOLIDATION FOR CONTINUAL PERSONALIZATION FROM BEHAVIORAL FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual personalization is often studied under idealized assumptions with explicit preference signals or offline task boundaries. In realistic deployment, however, systems must learn from streaming user interactions, where preferences are implicit and change points are never annotated. A central difficulty is deciding when recent adaptation should alter the model used for subsequent interactions: predictive mismatch may reflect a persistent preference shift, a temporary deviation, or a difficult input. We introduce Stable-and-Fast (SAF), a bounded state personalization method that separates continuous adaptation from selective consolidation. A Stable adapter serves predictions, while a Fast adapter learns from each revealed interaction. Before updating on the current feedback, SAF compares their predictive losses and accumulates relative evidence to consolidate Fast into Stable or reset Fast. We formulate this evidence as a length-normalized predictive log ratio and characterize its interpretation and limits under an evolving Fast adapter. We evaluate under a deployment-like streaming protocol on four LaMP derived tasks with 100 users per task, where SAF achieves good result compared to SOTA methods. These results establish selective consolidation as a distinct control decision for continual personalization in realistic streaming deployments.

1 INTRODUCTION

Language models increasingly support tasks that depend on individual users, including recommendation, writing assistance, and decision support. In these settings, a useful response must reflect both the input and the user's preferences. A headline generator may need to follow a user's writing style, while a rating predictor must account for how that user evaluates products. Existing personalization approaches incorporate user information through retrieved histories (Mysore et al., 2024), textual profiles (Zhang, 2024), or user-specific parameters (Tan et al., 2024b;a). LaMP provides a benchmark for studying personalized classification and generation, including ordinal product rating (Salemi et al., 2024). However, a user's needs and preferences can evolve as interactions accumulate (Jiang et al., 2025), making personalization based on an initial history less representative over time. This motivates *continual personalization*, where a model learns from ongoing user interactions to adapt to evolving needs and preferences.

Figure 1 provides a simple illustration of this temporal contrast. A static user-specific LoRA retains a preference representation learned from earlier interactions, whereas continual personalization updates that representation as later interactions provide evidence of changing interests. The shift from coffee and technology to tea and outdoor activities illustrates how an earlier representation can become misaligned with more recent behavior.

Continual personalization introduces a challenge beyond acquiring new information: *deciding how recent learning should affect future predictions*. A new interaction may reflect a persistent change in user needs, but it may also arise from a temporary context or an unusually difficult input. When one adapter is used for both online learning and prediction, each parameter update directly changes the model that serves subsequent requests. Frequent updates can help the model respond quickly, but they can also make its behavior sensitive to short-term variation. Restricting updates can preserve stability, yet may prevent the model from learning useful information. This tension raises a question: *how can learning continue while changes to the serving model are controlled separately?*

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

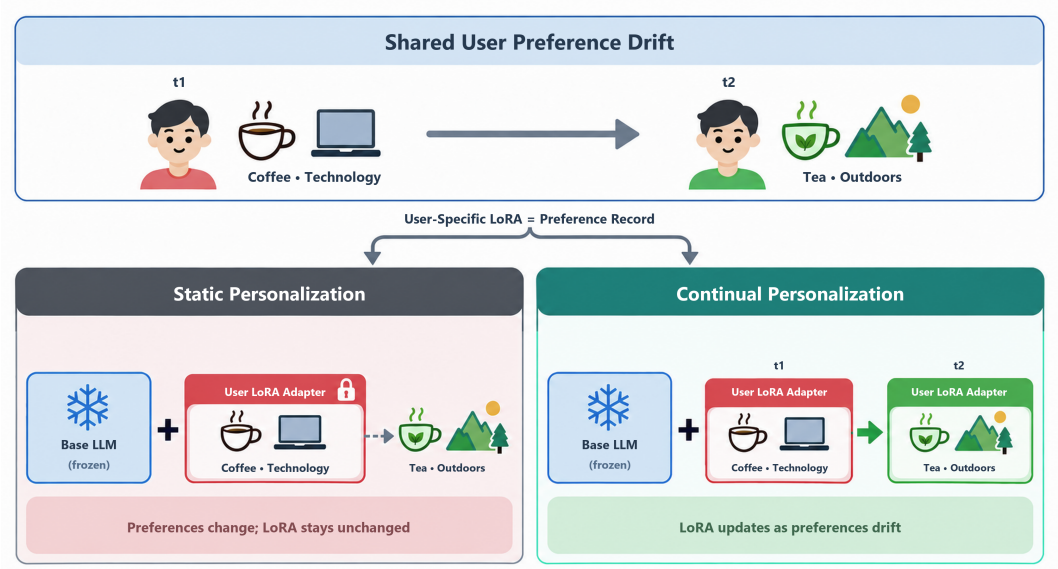


Figure 1: **Preference drift can make a static personalization record stale.**The schematic shows observed interests shifting from coffee and technology at t_1 to tea and outdoor activities at t_2, creating a mismatch between the user's later behavior and the earlier preference record.

Recent work addresses evolving preferences through different forms of feedback and adaptation. NS-DPO learns from timestamped preference pairs to account for preference drift (Son et al., 2025). Amulet steers generation using explicit preference prompts at inference time (Zhang et al., 2025). SPRInG combines selective parameter updates with replay and retrieval to support continual personalization (Kim & Kim, 2026). *These methods highlight the importance of deciding how new user information influences a model*. We focus on a complementary decision: when an updated parameter state should replace the state used for prediction. Selecting feedback for training and selecting a learned state for deployment serve different purposes. An interaction can contribute to learning without immediately changing the model exposed to the user.

We study continual personalization from behavioral feedback, such as ratings, tags, and user-authored text. Such feedback provides supervision for adaptation, but its implications for future interactions remain uncertain. Our focus is how to use this evidence to decide when learned updates should change the serving model. We introduce *selective consolidation* to make this decision explicit. We develop and evaluate the mechanism without storing raw interaction histories, targeting settings where retaining and reusing past user data is restricted. This setting allows us to examine consolidation using the current feedback and retained model states.

We implement this idea in Stable-and-Fast (SAF), using two user-specific LoRA adapters (Hu et al., 2022) over a shared, frozen task model. Stable produces the predictions used during interaction, while Fast learns from every revealed example. After feedback arrives, SAF compares the two adapters' losses before either learns from that example. Their loss difference measures Fast's relative predictive advantage on the same observation and admits a length-normalized predictive log-ratio interpretation. A controller accumulates this evidence across events. Sufficient accumulated advantage triggers copying Fast into Stable, whereas a sustained lack of advantage can instead reset Fast from Stable. Fast then trains on the current feedback. This separation allows continuous learning and selective changes to the serving state, while retaining only two adapters and a small set of controller variables (Figure 3). We evaluate SAF on four LaMP-derived tasks covering 400 user-task streams, measuring both online prediction quality and final-query performance. Compared with a task-aware selective-update baseline, SAF improves online performance on both generation tasks and final-query performance on three of the four tasks. Online performance declines on the two discrete-output tasks, indicating that the benefits of selective consolidation depend on the task.

Our contributions are summarized as follows:

(1) We formulate continual personalization from sequential behavioral feedback under a zero-replay constraint, making adaptation and consolidation explicit, separate decisions.

(2) We propose SAF, a dual-adapter method that uses accumulated pre-update predictive differences to control transfers between a continuously trained learner and a selectively updated serving model.
(3) We evaluate SAF across four personalization tasks, separating online performance from final-query performance and analyzing the task-dependent effects and computational costs of selective consolidation.

2 Related Work

Personalization with histories and adapters. LaMP evaluates personalization across classification, regression, and generation tasks (Salemi et al., 2024). OPPU uses personalized parameter-efficient fine-tuning and can combine parametric personalization with retrieved histories (Tan et al., 2024b). Building on their evidence for user-specific modeling, SAF uses LoRA (Hu et al., 2022) and adds an explicit consolidation decision that controls when adapted parameters become the serving state.
Continual personalization and selective adaptation. Kim & Kim (2026) propose SPRInG, which combines likelihood-based selective adaptation, residual replay, and retrieval-interpolated generation. Both SPRInG and SAF decide how behavioral evidence should influence personalization. SPRInG selects updates and combines them with replay and retrieval, whereas SAF continuously trains a Fast adapter and selectively transfers its state to a replay-free serving adapter. These designs expose complementary choices for selective continual personalization.
Online and test-time adaptation. Tent adapts a deployed model online using an unsupervised entropy objective (Wang et al., 2021). CoTTA studies continually changing target domains with teacher–student updates and restoration (Wang et al., 2022), while SAR analyzes instability under mixed shifts, small batches, and imbalanced streams (Niu et al., 2023). These studies show how online updates can accumulate errors and motivate explicit control of the serving state. SAF brings this stability problem to supervised personalized streams, using each revealed behavioral target to update user-specific LoRA parameters.
Learning from implicit and changing preferences. DRIFT extracts learning signals from user dissatisfaction in real-world interactions (Wang et al., 2026). Non-stationary DPO models temporally changing pairwise preferences and discounts older observations (Son et al., 2025). These methods design feedback signals and adaptation objectives for changing preferences. SAF contributes a complementary consolidation controller: its current training objective uses the revealed task target, while the controller determines when the learned adapter state should serve subsequent interactions. This decomposition connects feedback extraction, preference adaptation, and consolidation as distinct components of a continual personalization system.
Sequential evidence. Our reflected accumulation rule is inspired by cumulative-sum procedures (Page, 1954). SAF adapts this structure to evolving predictors through length-normalized, upper-capped increments computed on changing inputs. The resulting statistic summarizes sequential predictive advantage and controls consolidation, probabilistic change-detection guarantees would require an additional calibrated data model.

3 Continual Personalization: Problem Setting and Motivation

3.1 Problem Definition

Continual personalization aims to adapt a language model to a user's evolving needs and preferences through ongoing interactions. The model learns from observed user behavior to improve predictions in subsequent interactions. We consider behavioral feedback such as ratings, tags, and user-authored text. This feedback provides task supervision, while the underlying preferences and their change points remain unobserved.

For each user u, let $\{(x_{u,t}, y_{u,t})\}_{t=1}^{T_u}$ denote an ordered interaction stream, where $x_{u,t}$ is the input and $y_{u,t}$ is the corresponding feedback. At event t, the model predicts $\hat{y}_{u,t}$ using the current input and its state from previous interactions. It then observes $y_{u,t}$ and may update its state for future predictions. The conditional behavior $P_{u,t}(y \mid x)$ may change throughout the stream, requiring adaptation over time. In our setting, information from past interactions is retained in the model state. Raw inputs and feedback are discarded after processing.

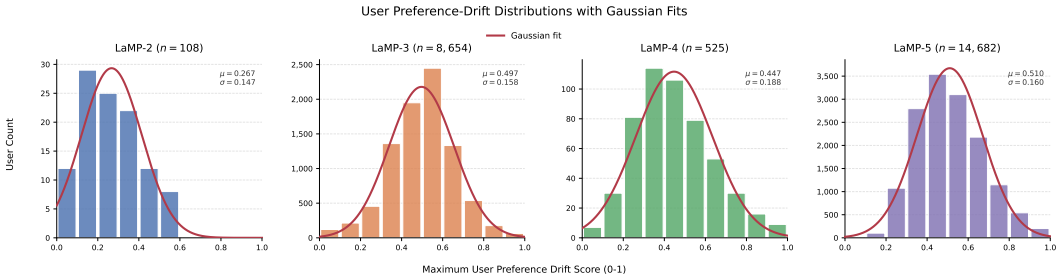


Figure 2: **User preference-drift intensity is broadly distributed across tasks.** Each value is the maximum distance D_u between a user's sliding-window preference centers.

3.2 Motivation for Selective Consolidation

Continual personalization involves two decisions that are coupled in a conventional single-adapter system: learning from newly revealed behavior and changing the model that serves subsequent predictions. Updating the same serving adapter after every event makes these decisions identical. Selective consolidation instead asks whether recent adaptation has accumulated enough predictive support to alter the serving state.

Users exhibit markedly different amounts of behavioral variation over time. To quantify this heterogeneity, we sort each user's profile chronologically and divide it into overlapping sliding windows of eight interactions with stride four. For LaMP-2/4/5, each window center $c_{u,a}$ is an embedding-based summary that combines the interaction text with the task-specific user response, such as a selected tag or user-authored text. For LaMP-3, the center is the empirical ordinal distribution over ratings 1–5, avoiding the treatment of rating values as ordinary text. Sentence embeddings with cosine similarity provide a precedent for the semantic text comparison (Reimers & Gurevych, 2019); Wasserstein distance provides a distributional comparison for ordinal ratings (Peyré & Cuturi, 2019). We define the drift intensity of user u as

$$D_u = \max_{(a,b)\in\mathcal{P}_u} d(c_{u,a}, c_{u,b}), \tag{1}$$

where $\mathcal{P}_u$ contains all valid pairs of windows for user u. The distance d is semantic cosine distance for LaMP-2/4/5 and normalized Wasserstein distance between rating distributions for LaMP-3.

Figure 2 shows that drift intensity varies substantially both within and across tasks, making a single unconditional update schedule unlikely to suit every user. The score operationalizes temporal behavioral variation, and thresholds such as $D_u > 0.2$ identify users with stronger observed changes across interaction windows.

4 Stable-and-Fast Personalization

Figure 3 connects the shared backbone and two user adapters to the evidence controller and their evolution across events. The two model drawings represent parameter configurations of one frozen backbone. After bootstrap, each event follows the same order: predict, reveal feedback, score, decide whether to copy, train Fast, and discard the raw event.

4.1 Separate States for Serving and Learning

Stable and Fast correspond to two preference regimes. Stable represents long-lived or slowly changing preferences and is the only state used for serving predictions. Fast represents recent and potentially drifting preferences. A commit promotes Fast into Stable when accumulated evidence supports a change; a reset returns Fast to Stable when recent variation does not accumulate into a persistent advantage.

Let θ_T be a task-adapted backbone that remains frozen. A LoRA update has the form $\Delta W = (\alpha/r)BA$, where r is the adapter rank (Hu et al., 2022). SAF maintains two sets of User-LoRA parameters, ϕ_t^S and ϕ_t^F, with predictive distributions p_{S_t} and p_{F_t}. Stable serves $\hat{y}_t$, while Fast adapts to recent feedback until consolidation.

Both adapters begin from the same initialization. For the first B events, the system predicts with Stable, trains Stable on the revealed event, and copies its parameters to Fast. The controller is inactive during this bootstrap. Thereafter, Fast is trained once on each revealed event, while Stable changes only through consolidation. Reset and commit operations copy adapter parameters directly.

4.2 Relative evidence from pre-update feedback

Let $y_t = (y_{t,1}, \ldots, y_{t,L_t})$ denote the scored feedback tokens after the common tokenization and loss mask, including the scored end-of-sequence token. For $A \in \{S_t, F_t\}$, define

$$\ell_t(A) = -\frac{1}{L_t} \sum_{j=1}^{L_t} \log p_A(y_{t,j} \mid x_t, y_{t,<j}), \qquad d_t = \ell_t(S_t) - \ell_t(F_t). \tag{2}$$

Teacher forcing uses the current feedback prefix $y_{t,<j}$. The two personalized losses are computed before training on (x_t, y_t). A positive d_t means that the pre-update Fast model explains this feedback better than Stable.

Predictive log-ratio interpretation. For the same scored token set, define $q_A(y_t \mid x_t) = \prod_{j=1}^{L_t} p_A(y_{t,j} \mid x_t, y_{t,<j})$. Then

$$d_t = \frac{1}{L_t} \log \frac{q_{F_t}(y_t \mid x_t)}{q_{S_t}(y_t \mid x_t)}. \tag{3}$$

This identity compares both adapters on the same observation. It also has a task-relative interpretation: if $a_t(A) = \ell_t(T) - \ell_t(A)$ is an adapter's gain over the frozen task model, then $d_t = a_t(F_t) - a_t(S_t)$. The task-model loss cancels exactly. In the implementation it is recorded for diagnostics.

Accumulation and decisions. Within a controller episode, let E be accumulated evidence, n the number of scored events, and r_0 the length of the current non-advantage streak. Given slack $s \geq 0$, positive cap M, threshold $h > 0$, minimum count K, and reset patience P, compute

$$v_t = \min\{d_t - s, M\}, \qquad E \leftarrow \max\{0, E + v_t\}, \tag{4}$$

$$n \leftarrow n + 1, \qquad r_0 \leftarrow \begin{cases} r_0 + 1, & d_t \leq s, \\ 0, & d_t > s. \end{cases} \tag{5}$$

A commit occurs if $n \geq K$ and $E \geq h$. Otherwise, a reset occurs if $E = 0$ and $r_0 \geq P$. A commit copies pre-update Fast into Stable, a reset copies Stable into Fast. Either action clears (E, n, r_0). Fast then trains on the current feedback. If neither action occurs, the adapters are not copied and Fast still trains. Thus, the current feedback may influence the decision to copy an existing model, but the model being copied has not been trained on that feedback.

Figure 3(b) uses time-indexed controller states. Here, $(\widetilde{E}_t, \widetilde{n}_t, \widetilde{r}_t)$ denotes (E, n, r_0) after incorporating event t and before the copy decision. Panel (c) shows why Fast can differ from Stable immediately after a commit: only Fast then learns the current event.

The upper cap limits positive evidence from one event, while negative increments remain uncapped. The variable K counts episode observations, not consecutive wins. With our defaults $h = 0.5$ and $M = 1$, one favorable increment can cross the evidence threshold once the observation-count requirement is met.

4.3 Interpretation, causality, and cost

Between episode resets, the reflected statistic is the largest positive suffix sum:

$$E_t = \max\left\{0, \max_{k=\tau+1,\ldots,t} \sum_{i=k}^{t} v_i\right\}, \tag{6}$$

where $E_\tau = 0$ starts the episode. This standard identity explains why earlier unfavorable observations need not suppress later evidence indefinitely.

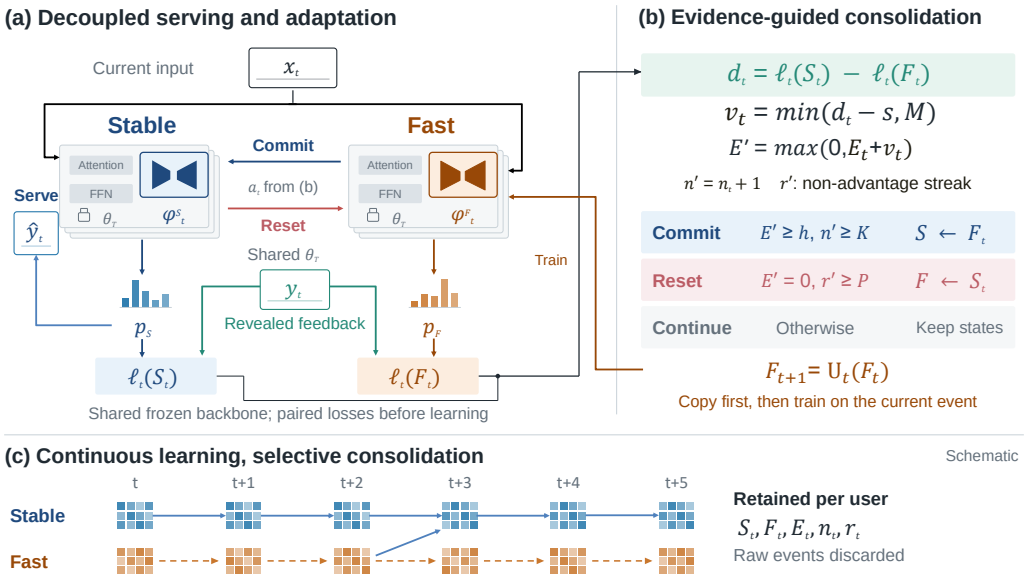


Figure 3: **SAF: evidence-guided consolidation for continual personalization.** (a) Stable and Fast are two User-LoRA configurations of the same frozen task-adapted backbone θ_T. Stable alone serves predictions. After feedback is revealed, both configurations score the same target tokens before learning, with teacher forcing on the current feedback prefix. Colored solid arrows copy only the user-adapter parameters; the dashed orange path trains Fast after the decision. (b) Relative evidence controls commit, reset, or continued adaptation. $\widetilde{n}_t$ counts episode observations and $\widetilde{r}_t$ the non-advantage streak. $\overline{F}_t$ denotes Fast after any reset, and $\mathcal{U}_t$ trains it on (x_t, y_t). Both copy actions clear the controller statistics. (c) Fast changes after each event; a commit changes Stable for the next event. Distributions and parameter matrices are schematic. The figure depicts post-bootstrap operation.

Fast changes after every event, so Eq. (6) aggregates evidence for an *adaptive sequence* of alternatives. It is not a held-out evaluation of one fixed candidate, and cannot by itself certify the checkpoint eventually committed. Likewise, length normalization, clipping, and model adaptation mean that exponentiating E_t does not produce a calibrated likelihood-ratio test. These are limits of the present controller, not additional guarantees assumed by the method.

The controller retains two adapter states and three scalars, independently of stream length for fixed tensor shapes and scalar representations. Shared backbone parameters are excluded from the user-specific count. SAF performs a training call on every event, including bootstrap events; selective consolidation does not imply selective backward computation. Scoring two personalized models also adds work relative to a single-model controller. We report update calls and consolidation counts separately from predictive metrics, without translating them into unmeasured FLOPs or wall-clock savings.

5 Experimental

5.1 Experimental Details

Datasets. We evaluate four LaMP-derived tasks: movie tagging (LaMP-2), product rating (LaMP-3), news headline generation (LaMP-4), and scholarly title generation (LaMP-5). Each task uses a fixed 100-user cohort selected by the existing drift-stratified construction. This is not a random population sample. Every profile item is first predicted and then revealed for possible adaptation. The cohorts contain 54,782 profile events in total. Dates are nondecreasing within each stream, but many adjacent records share a timestamp. The retained within-timestamp ordering is therefore part of the evaluation protocol. Strata and timestamps do not provide ground-truth latent preference-change labels. Table 1 reports the evaluated scope.

Table 1: Evaluated cohorts and task training data. Each task we have chosen 100 evaluation users. The task-user pool is counted before exact-sample exclusions, while task samples are retained supervised examples used to train the shared Task-LoRA. Final queries are evaluated after profile stream.

Task	Profile events	Final queries	Task-user pool	Task samples
Movie tagging	5,414	2,717	821	1,281
Product rating	28,372	100	1,200	12,000
News headline	12,246	2,786	1,085	12,000
Scholarly title	8,750	100	1,200	12,000

Paired user bootstrap. For each task, let m_u^A be a user's event-averaged score for method A. The paired contrast is $\Delta = U^{-1} \sum_u (m_u^{\text{SAF}} - m_u^{\text{Main}})$. Each bootstrap draw samples $U = 100$ user indices with replacement and averages the paired differences at those indices. Resampling whole users rather than individual events follows the cluster-bootstrap principle (Field & Welsh, 2007). We use 5,000 draws and the 2.5th and 97.5th percentiles of the bootstrap distribution (Efron, 1987), with resampling seed 20260909. Pairing preserves the evaluation unit and each user's within-stream dependence. It does not account for variability from new model-training seeds, unseen cohorts, or decoding configurations.

Effect of preference drift. Figure 4 reports OPPU performance across the preference-drift ranges defined by Equation 1. On LaMP-2, both Accuracy and Macro-F1 decline sharply in the highest-drift range. LaMP-4 shows a similar pattern for ROUGE-1 and ROUGE-L, with both metrics reaching their lowest values in the 0.5–0.6 range. The intermediate ranges are not strictly monotonic, so this analysis does not establish a causal relationship between drift and performance. It nevertheless suggests that static profile-based personalization can become less reliable under stronger behavioral variation, motivating evaluation in a continual setting.

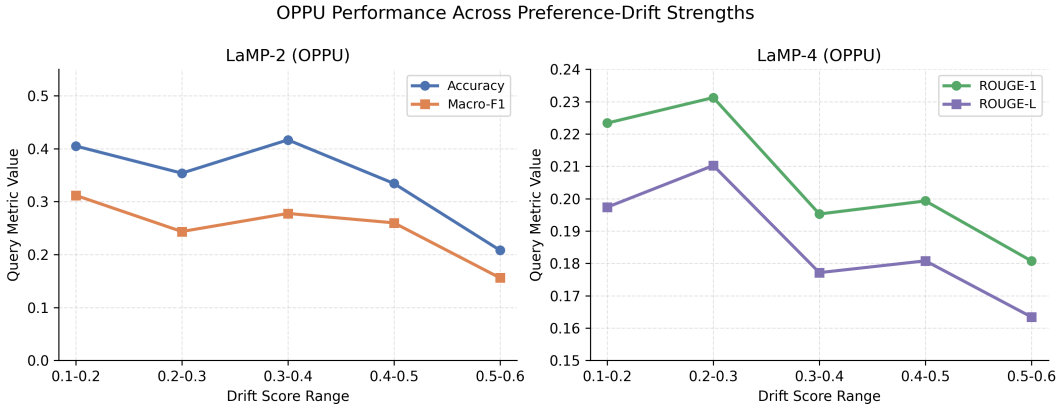


Figure 4: **OPPU performance across preference-drift strengths.** Query metrics are grouped by user drift-score range for LaMP-2 and LaMP-4.

Compared personalization methods. OPPU learns a separate parameter-efficient module from each user's history (Tan et al., 2024b). C-BPO contrasts target-user observations with auxiliary data from other users and corrects for overlapping preferences through a positive–unlabeled objective (Ma et al., 2026). PURPLE learns a contextual-bandit policy for selecting and ordering the historical records supplied to a retrieval-augmented LLM (Du et al., 2026). These baselines respectively represent per-user parametric adaptation, cross-user binary-feedback learning, and retrieval-based profile optimization.

Implementation details. SAF uses the Llama 2 7B backbone (Touvron et al., 2023) with a shared Task-LoRA merged before user adaptation. Task training uses separate users and excludes exact supervised-sample overlap with the evaluation streams before sampling. We verified the recorded exclusion-file hashes and reconstructed this filtering: no exact-key or user overlap remains under the implemented matching rules. This check does not detect semantic near-duplicates. The User-LoRA has rank 8, scale 8, dropout 0.05, and targets the query and value projections. Each adaptation call trains for one epoch on the current event at learning rate 10^{-4}. SAF uses $(B, h, s, M, K, P) = (4, 0.5, 0, 1, 2, 4)$ across all reported runs.

Table 2: **Main experimental results across four LaMP tasks.**

Method	LaMP-2		LaMP-3		LaMP-4		LaMP-5	
	A (%)↑	F1 (%)↑	MAE↓	RMSE↓	R1 (%)↑	RL (%)↑	R1 (%)↑	RL (%)↑
RAG	45.88 ± 0.24	31.32 ± 0.11	0.21 ± 0.009	0.506 ± 0.018	20.94 ± 0.04	19.08 ± 0.13	45.06 ± 0.26	40.49 ± 0.13
OPPU	51.34 ± 0.17	34.71 ± 0.15	$\mathbf{0.16 \pm 0.006}$	$\mathbf{0.45 \pm 0.025}$	20.75 ± 0.10	18.66 ± 0.06	49.50 ± 0.17	45.08 ± 0.18
C-BPO	20.98 ± 0.06	15.14 ± 0.05	0.76 ± 0.044	1.46 ± 0.058	13.56 ± 0.02	11.95 ± 0.03	42.12 ± 0.21	37.54 ± 0.13
PURPLE	29.00 ± 0.17	29.11 ± 0.13	0.25 ± 0.013	0.59 ± 0.019	14.41 ± 0.13	13.12 ± 0.12	38.60 ± 0.16	35.77 ± 0.21
Ours	$\mathbf{57.34 \pm 0.26}$	$\mathbf{42.67 \pm 0.17}$	0.17 ± 0.010	0.46 ± 0.021	$\mathbf{22.80 \pm 0.05}$	$\mathbf{20.73 \pm 0.11}$	$\mathbf{50.76 \pm 0.21}$	$\mathbf{46.67 \pm 0.18}$

Evaluation target. For an event-level score m and U evaluated users, we report

$$\overline{m}_{\text{online}} = \frac{1}{U} \sum_{u=1}^{U} \frac{1}{T_u} \sum_{t=1}^{T_u} m(\hat{y}_{u,t}, y_{u,t}). \tag{7}$$

This gives each user equal weight and preserves the causal, event-by-event evaluation target.

Evaluation metrics. Table 2 reports Accuracy and Macro-F1 for LaMP-2, MAE and RMSE for LaMP-3, and ROUGE-1/ROUGE-L F1 (Lin, 2004) for LaMP-4 and LaMP-5. Accuracy, Macro-F1, and ROUGE are expressed as percentages. We use user-averaged accuracy, MAE, and ROUGE-L, following Equation 7. We recompute these paired metrics from the existing event predictions, matching user identities, event order, and reference outputs across methods. Paired 95% intervals use 5,000 bootstrap resamples of users, keeping each user's events together (Field & Welsh, 2007); interval endpoints are the empirical 2.5th and 97.5th percentiles (Efron, 1987). These intervals describe variation across the fixed evaluated cohort conditional on one training seed (42). They do not estimate training-seed uncertainty. The paired methods are not matched for adaptation calls, and some inference batch sizes differ, so their observed effects are not attributed solely to the controller.

5.2 Main results across four LaMP tasks

The drift-stratified evaluation follows the preference-drift definition in Equation 1. LaMP-2/3 are treated as classification/discrete-output tasks, and LaMP-4/5 as generation tasks. The drift bins in Figure 5 use the same definition to show how continual adaptation and the static OPPU reference behave as drift increases.

Table 2 shows that SAF Full obtains the best result on six of eight metrics. Relative to OPPU, the strongest baseline in every reported column, SAF improves LaMP-2 Accuracy and Macro-F1 by 6.00 and 7.96 percentage points, respectively. It also improves ROUGE-1/ROUGE-L by 2.05/2.07 points on LaMP-4 and 1.26/1.59 points on LaMP-5. LaMP-3 is the exception: OPPU reduces MAE by 0.010 and RMSE by 0.012 relative to SAF. The main comparison therefore favors SAF on movie tagging and both generation tasks, but not on product rating.

Beyond aggregate performance, we examine how each method behaves as user preference drift increases. We group LaMP-2 users by drift-score and compare SAF with OPPU and RAG in Figure 5.

As shown in Figure 5, on the representative classification task LaMP-2, the Accuracy and Macro-F1 of the static OPPU baseline decrease substantially as user preference drift becomes more pronounced. In contrast, our SAF method maintains competitive Accuracy across different drift levels, remaining between 0.42 and 0.51 in the first four drift intervals and reaching 0.71 in the highest-drift interval. More importantly, the Macro-F1 of SAF remains within a narrow range of 0.38–0.42, with an overall variation of less than 0.04. While OPPU and RAG exhibit noticeable degradation or fluctuations as drift increases, SAF delivers consistently stable performance. These results indicate that its dynamic adaptation mechanism effectively mitigates performance degradation caused by evolving user preferences and provides more robust classification across different drift levels.

5.3 Ablation study

Table 3 shows that the full controller achieves the best result on five of eight metrics, including LaMP-2 accuracy, both LaMP-3 error metrics, and both LaMP-4 ROUGE metrics. Instantaneous decisions underperform the full method on every reported metric, supporting the use of accumulated rather than one-step evidence. Direct-Stable is also weaker on six metrics, although it improves both LaMP-5 scores. Removing resets yields the best LaMP-2 Macro-F1 and LaMP-5 ROUGE scores,

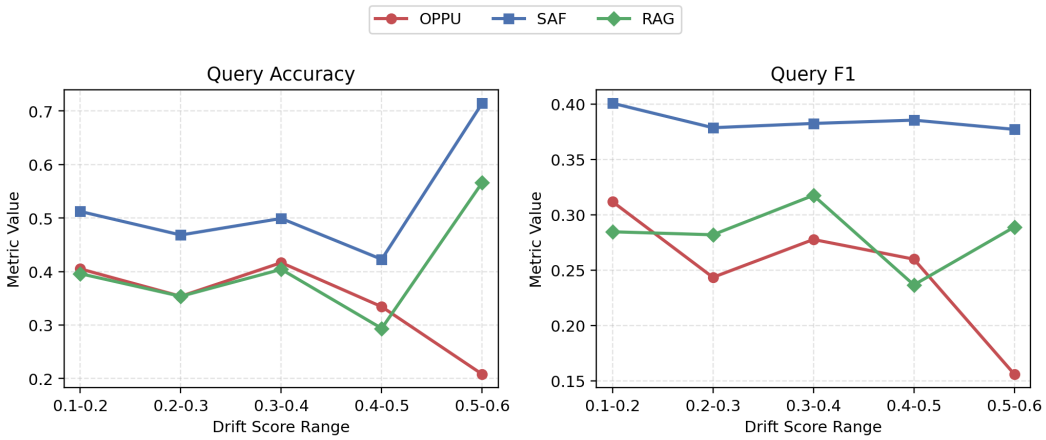


Figure 5: **LaMP-2 performance across preference-drift levels.** The left and right panels report Accuracy and Macro-F1, respectively, for SAF, OPPU, and RAG across drift-score intervals.

Table 3: **Ablation results across four LaMP tasks.**

Method	LaMP-2		LaMP-3		LaMP-4		LaMP-5	
	A (%)↑	F1 (%)↑	MAE↓	RMSE↓	R1 (%)↑	RL (%)↑	R1 (%)↑	RL (%)↑
Full	**57.34**	42.67	**0.170**	**0.459**	**22.80**	**20.73**	50.76	46.67
Direct-Stable	56.68	42.32	0.190	0.479	21.24	19.08	51.57	46.98
Instantaneous	54.99	41.22	0.200	0.490	20.85	18.89	49.09	44.84
No-Reset	57.12	**42.70**	0.200	0.490	21.86	19.83	**51.81**	**47.01**

but degrades LaMP-3 and LaMP-4. Thus, accumulation and serving–learning separation are broadly useful in these runs, whereas the benefit of reset behavior remains task dependent.

5.4 Control behavior and accounting

Across 53,182 post-bootstrap events, SAF performs 1,881 commits (3.54%), directly showing sparse changes to the serving adapter despite continuous Fast learning. These counts measure control activity, not change-detection accuracy. SAF makes 3.67 times as many event-level training calls as OPPU, with additional scoring overhead. Because call costs depend on sequence length and no matched runtime or FLOP measurements are available, the observed gains establish neither a compute advantage nor the contribution of consolidation independently of extra adaptation.

6 Conclusion

SAF separates continuous behavioral adaptation from selective changes to the serving model. Its relative-evidence controller admits a predictive log-ratio interpretation while retaining bounded user state. In the main comparison, SAF achieves the best result on six of eight metrics, leading OPPU, C-BPO, and PURPLE on LaMP-2 and both generation tasks, but trailing OPPU on LaMP-3. Ablations favor accumulated over instantaneous evidence, though reset behavior remains task dependent.

The evaluation is limited to four fixed cohorts and one backbone. SAF also uses 3.67× more event-level training calls than OPPU and two additional scoring passes, so a matched-budget single-adapter control is needed to isolate consolidation from extra learning. Decoding batch differences and coarse timestamps add further confounds. Without ground-truth change points, predictive gains cannot establish preference-change precision or delay, and the evolving Fast adapter precludes certification of one fixed candidate. Zero replay is an access constraint, not a privacy guarantee or evidence of superiority over bounded replay and retrieval. Controlled transient-versus-persistent streams and action-level evaluation of committed snapshots are needed to determine when selective consolidation prevents harmful long-term writes while remaining responsive to persistent changes.

AI USE STATEMENT

Generative AI tools assisted with the interpretation of existing results, manuscript editing, literature checking and figure creation. They did not generate the experimental model predictions reported here.

REPRODUCIBILITY STATEMENT

Section 4 and Appendix A specify the event ordering and controller. Section 5 and Appendix C describe data access, model settings, and known configuration differences. Appendix D specifies metric aggregation and uncertainty estimation; Appendix E describes the early-prototype diagnostic and its provenance limits. The current results are a single-seed analysis of saved predictions; no independent training replication is claimed.

REFERENCES

Linfeng Du, Ye Yuan, Zichen Zhao, Fuyuan Lyu, Emiliano Penaloza, Xiuying Chen, Zipeng Sun, Jikun Kang, Laurent Charlin, Xue Liu, and Haolun Wu. Optimizing user profiles via contextual bandits for retrieval-augmented LLM personalization. In *Proceedings of ACL*, pp. 31803–31817, 2026. doi: 10.18653/v1/2026.acl-long.1467. URL https://aclanthology.org/2026.acl-long.1467/.

Bradley Efron. Better bootstrap confidence intervals. *Journal of the American Statistical Association*, 82(397):171–185, 1987. doi: 10.1080/01621459.1987.10478410. URL https://doi.org/10.1080/01621459.1987.10478410.

C. A. Field and A. H. Welsh. Bootstrapping clustered data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(3):369–390, 2007. doi: 10.1111/j.1467-9868.2007.00593.x. URL https://doi.org/10.1111/j.1467-9868.2007.00593.x.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2106.09685.

Bowen Jiang, Zhuoqun Hao, Young-Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo J. Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. *arXiv preprint arXiv:2504.14225*, 2025. URL https://arxiv.org/abs/2504.14225.

Seoyeon Kim and Jaehyung Kim. SPRInG: Continual LLM personalization via selective parametric adaptation and retrieval-interpolated generation. *arXiv preprint arXiv:2601.09974*, 2026. URL https://arxiv.org/abs/2601.09974.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, 2004. URL https://aclanthology.org/W04-1013/.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Xilai Ma, Liye Zhao, Weijun Yao, Haibing Di, Wenya Wang, and Jing Li. Personalizing LLMs with binary feedback: A preference-calibrated optimization framework. In *Proceedings of ACL*, pp. 26539–26555, 2026. doi: 10.18653/v1/2026.acl-long.1222. URL https://aclanthology.org/2026.acl-long.1222/.

Sheshera Mysore, Zhuoran Lu, Mengting Wan, Longqi Yang, Bahareh Sarrafzadeh, Steve Menezes, Tina Baghaee, Emmanuel Barajas Gonzalez, Jennifer Neville, and Tara Safavi. Pearl: Personalizing large language model writing assistants with generation-calibrated retrievers. In *Proceedings of CustomNLP4U*, pp. 198–219, 2024. doi: 10.18653/v1/2024.customnlp4u-1.16. URL https://aclanthology.org/2024.customnlp4u-1.16/.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=g2YraF75Tj.

E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1–2):100–115, 1954. doi: 10.1093/biomet/41.1-2.100.

Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5–6):355–607, 2019. doi: 10.1561/2200000073. URL https://www.nowpublishers.com/article/Details/MAL-073.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3982–3992, 2019. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. LaMP: When large language models meet personalization. In *Proceedings of ACL*, pp. 7370–7392, 2024. doi: 10.18653/v1/2024.acl-long.399. URL https://aclanthology.org/2024.acl-long.399/.

Seongho Son, William Bankes, Sayak Ray Chowdhury, Brooks Paige, and Ilija Bogunovic. Right now, wrong then: Non-stationary direct preference optimization under preference drift. In *Proceedings of ICML*, volume 267 of *Proceedings of Machine Learning Research*, pp. 56173–56203, 2025. URL https://proceedings.mlr.press/v267/son25a.html.

Zhaoxuan Tan, Zheyuan Liu, and Meng Jiang. Personalized pieces: Efficient personalized large language models through collaborative efforts. In *Proceedings of EMNLP*, pp. 6459–6475, 2024a. doi: 10.18653/v1/2024.emnlp-main.371. URL https://aclanthology.org/2024.emnlp-main.371/.

Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. Democratizing large language models via personalized parameter-efficient fine-tuning. In *Proceedings of EMNLP*, 2024b. doi: 10.18653/v1/2024.emnlp-main.372. URL https://aclanthology.org/2024.emnlp-main.372/.

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

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2006.10726.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7201–7211, 2022. URL https://arxiv.org/abs/2203.13591.

Yifan Wang, Bolian Li, Junlin Wu, Zhaoxuan Tan, Zheli Liu, Ruqi Zhang, Ananth Grama, and Qingkai Zeng. DRIFT: Learning from abundant user dissatisfaction in real-world preference learning. In *International Conference on Learning Representations*, pp. 120457–120479, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/c40bed606c51c8e827c1ba75aa2da054-Paper-Conference.pdf.

Jiarui Zhang. Guided profile generation improves personalization with large language models. In *Findings of EMNLP*, pp. 4005–4016, 2024. doi: 10.18653/v1/2024.findings-emnlp.231. URL https://aclanthology.org/2024.findings-emnlp.231/.

Zhaowei Zhang, Fengshuo Bai, Qizhi Chen, Chengdong Ma, Mingzhi Wang, Haoran Sun, Zilong Zheng, and Yaodong Yang. Amulet: Realignment during test time for personalized preference adaptation of LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=f9w89OY2cp.

A CONTROLLER SPECIFICATION

The following pseudocode states the implemented control order. TRAIN means one event-level training call, which can contain multiple optimizer steps if tokenization produces multiple training examples. Only the two adapter states and controller scalars persist in the online algorithm.

Algorithm 1: Stable-and-Fast personalization for one user

Input: frozen task model; stream; B, h, s, M, K, P.
Initialize: $S \leftarrow S_0$, $F \leftarrow S_0$, $(E, n, r_0) \leftarrow (0, 0, 0)$.
For each event $t = 1, \ldots, T$:
  Observe x_t and emit a prediction from S.
  Receive y_t; score pre-update S, F (and task model for diagnostics).
  If $t \leq B$:
    $S \leftarrow \text{TRAIN}(S; x_t, y_t)$; $F \leftarrow S$.
    Keep $(E, n, r_0) = (0, 0, 0)$; continue to the next event.
  $d \leftarrow \ell_t(S) - \ell_t(F)$; $E \leftarrow \max\{0, E + \min(d - s, M)\}$.
  $n \leftarrow n + 1$; $r_0 \leftarrow r_0 + 1$ if $d \leq s$, otherwise $r_0 \leftarrow 0$.
  If $n \geq K$ and $E \geq h$:
    $S \leftarrow F$; $(E, n, r_0) \leftarrow (0, 0, 0)$ [commit]
  Else if $E = 0$ and $r_0 \geq P$:
    $F \leftarrow S$; $(E, n, r_0) \leftarrow (0, 0, 0)$ [reset]
  $F \leftarrow \text{TRAIN}(F; x_t, y_t)$.
  Discard raw (x_t, y_t) from the learner's accessible state.

The first B predictions are evaluated before each corresponding bootstrap update. After bootstrap, a commit transfers Fast *before* its update on the current feedback. The same event can therefore trigger consolidation without entering the transferred parameters through current-event training. A reset also precedes current-event training, so the next Fast state begins by adapting from Stable on that feedback. Commit takes precedence over reset. Parameter copies do not transfer an accumulated optimizer state: the implementation constructs a fresh optimizer for each adaptation call.

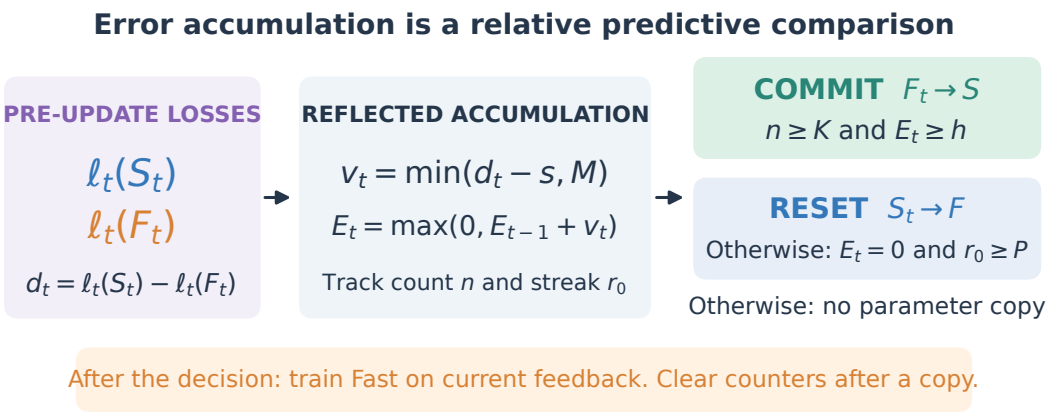


Figure 6: **Relative-evidence controller, shown separately.** Both losses are pre-update. Positive increments are capped; negative increments are not. A commit clears the controller and copies Fast to Stable. If commit does not fire, zero evidence together with a sufficiently long non-advantage streak triggers reset. Either decision is followed by training Fast on the revealed feedback.

B Predictive Identities and Their Scope

Log-ratio identity. Subtracting the two losses in Equation 2 gives

$$\begin{aligned}\ell_t(S_t) - \ell_t(F_t) &= \frac{1}{L_t}\sum_{j=1}^{L_t}\left[\log p_{F_t}(y_{t,j} \mid x_t, y_{t,<j}) - \log p_{S_t}(y_{t,j} \mid x_t, y_{t,<j})\right] \\ &= \frac{1}{L_t}\log\frac{q_{F_t}(y_t \mid x_t)}{q_{S_t}(y_t \mid x_t)}.\end{aligned}$$

The same token set and denominator must be used for both models. If targets are truncated or chunked, the identity applies to the scored conditional factors, not to an unobserved full response. The task-relative gain identity follows by cancellation of the common $\ell_t(T)$. It is an interpretation of the existing comparison, not a new reward function or regularizer.

Reflected-sum identity. The reflected recursion has the cumulative-sum form introduced by Page (1954); the identity below applies to the specific increments and episode resets used here. Let $E_\tau = 0$. For one event, $E_{\tau+1} = \max\{0, v_{\tau+1}\}$. Suppose Equation 6 holds at $t-1$. Substituting it into $E_t = \max\{0, E_{t-1} + v_t\}$ yields

$$E_t = \max\left\{0, v_t, \max_{k=\tau+1,\ldots,t-1}\sum_{i=k}^{t} v_i\right\},$$

which is Equation 6. Thus a threshold crossing means that at least one within-episode suffix has capped, slack-adjusted cumulative advantage at least h. It does not require every event in that suffix to favor Fast. Because $v_i \le M$, any such suffix contains at least $\lceil h/M \rceil$ events. The separate K condition counts all events since the episode began. It does not guarantee K positive increments. With the reported h and M, $\lceil h/M \rceil = 1$.

Adaptive alternatives and testing. The index F_t changes with the history, so the uncapped sum is a comparison of sequential predictors rather than repeated observations from one frozen Fast model. In addition, $q_{F_t}^{1/L_t}$ and $q_{S_t}^{1/L_t}$ need not be normalized distributions over complete outputs. The cap and slack further alter the statistic. Consequently, neither the log-ratio identity nor the reflected-sum identity establishes a false-alarm probability, detection-delay bound, or guarantee about future performance after a commit. Such guarantees would require an explicitly specified null model and a suitably calibrated procedure.

C Implementation and Data Details

Table 4: Shared adaptation settings and SAF controller defaults.

Setting	Value
Backbone	Llama 2 7B; BF16; no quantization
Shared Task-LoRA	Rank 8, scale 8; query/key/value/output projections
Task-LoRA training	Three epochs; seed 42; at most 12,000 samples
User-LoRA	Rank 8, scale 8; query/value projections; dropout .05
Per-event optimization	AdamW; learning rate 10^{-4}; one epoch
Batch / accumulation	1 / 1
Weight decay / gradient norm	.01 / .3
Learning-rate schedule	Constant; no warmup
Optimizer persistence	Reinitialized for each event-level training call
SAF bootstrap / threshold	$B = 4$, $h = .5$
SAF slack / cap	$s = 0$, $M = 1$
SAF minimum count / patience	$K = 2$, $P = 4$
CUSUM ref. confidence / threshold	.5 / 2

The shared backbone is Llama 2 7B (Touvron et al., 2023). The per-event optimizer is AdamW, which uses decoupled weight decay (Loshchilov & Hutter, 2019). For a 4,096-dimensional attention

projection, a rank-8 LoRA (Hu et al., 2022) contains $8(4096 + 4096)$ parameters. With two targeted projections across 32 layers, each User-LoRA contains 4,194,304 parameters. The two adapters together contain 8,388,608. This is an architecture-derived count of retained user parameters, not a measured GPU-memory footprint. Activations, temporary optimizer tensors, shared model weights, and evaluation infrastructure are additional costs.

Decoding and configuration differences. Discrete outputs use greedy decoding with a four-token generation cap. Generation tasks use sampling with temperature .1, top-k 10, and top-p .9. The token cap is 64 for stream predictions and 200 for final queries. The recorded token cutoffs are 384, 512, 256, and 512 for movie, product, news, and scholarly tasks, respectively. Inference batch sizes for SAF/Main/CUSUM are 64/64/4, 1/1/2, 32/2/2, and 1/1/1. Batch-size differences can affect numerics and sampled generation. They were not eliminated in the available runs. The shared task checkpoint and cohort hashes match across these three methods within each task.

Offline OPPU reference. The local reproduction starts from the task-adapted backbone and trains a rank-8, scale-8 user adapter on query/value projections for two full-profile epochs, with learning rate 10^{-4}, effective batch size 16, and token cutoff 2,048. Its inference batch sizes are 4, 2, 2, and 1 across the four tasks. Query decoding follows the matched-current-experiment option. The recorded preprocessing audit flags 15 fully truncated rating targets, and none on the other three tasks. These preprocessing and optimization differences further limit direct attribution of final-query differences to the online controller.

Cohort ordering. The numbers of adjacent equal-date pairs are 3,577/5,314 for movie tagging, 12,243/28,272 for product rating, 3,182/12,146 for news, and 6,808/8,650 for scholarly titles. No date inversion was found. Equal dates leave temporal order underdetermined, particularly for movie and scholarly tasks. All methods retain the same within-timestamp order, but this does not validate that order as the true interaction sequence.

Task-training overlap exclusions. We reconstruct the training-time exact-key filtering using the recorded input and exclusion-file hashes. It removes overlapping supervised query/profile samples, rather than treating a reused profile identifier alone as proof of duplicated supervision. Across the four tasks, the remaining user and exact supervised-key intersections are zero under this rule. The check concerns the implemented matching representation; paraphrases and other semantic overlaps may remain. Task samples are independent history-free examples: a training example does not concatenate the evaluation user's accumulated history.

Feedback interpretation. The LaMP task targets—ratings, tags, and user-authored outputs—provide task supervision (Salemi et al., 2024). They are behavioral observations from which personalized conditional behavior is inferred. They do not directly reveal a preference vector, change point, confidence in a preference, or reason for a change. In particular, these experiments do not evaluate detecting dissatisfaction in free-form dialogue or obtaining feedback from silence, clicks, or missing responses.

D Additional Metrics and Aggregation

Table 5: Observed control activity. Update counts are event-level training calls, not optimizer steps or FLOPs. SAF updates on every event; commits and resets exclude bootstrap copies.

Task	Main updates	SAF updates	SAF commits	SAF resets
Movie tagging	2,515	5,414	245	63
Product rating	5,926	28,372	1,102	678
News headline	4,416	12,246	286	74
Scholarly title	2,087	8,750	248	47
Total	14,944	54,782	1,881	862

Scoring and invalid outputs. Movie accuracy compares the normalized prediction and reference label. Product MAE uses the normalized numeric prediction, an invalid numeric output receives the error of the farther endpoint of the 1–5 rating scale, following the existing evaluator. Online ROUGE is recomputed as the mean of per-event F1 values within each user and then across users. Previously recorded pooled profile summaries use a different aggregation, including the existing bootstrap-based ROUGE aggregator.

Evidence boundaries. No new model predictions were generated for this analysis. The additional user averages and bootstrap intervals are computed from saved, aligned predictions. Control counts describe executed actions and cannot be interpreted as true or false detections without change annotations. The current results provide no direct test of whether a frozen candidate, a different preference-learning objective, or a context-conditioned detector would improve SAF.

E HISTORICAL SAF PROTOTYPE DIAGNOSTIC

Table 6 gives the saved query results underlying Within each bin, the static base, early SAF prototype, and historical MoE runs have matching user identities and query IDs in the same order. Users are disjoint across bins. We verified that the plotted numbers match the saved per-bin experiment summaries; no new model predictions were generated.

Table 6: Historical movie-tagging results. Accuracy and macro-F1 are pooled over final queries within each bin; gains are the early SAF prototype minus the static base in percentage points. These cohorts and this baseline are separate from the final SAF evaluation.

			Accuracy		Macro-F1		Gain (pp)	
Bin	Users	Queries	Base	Early SAF	Base	Early SAF	Acc.	F1
0.1–0.2	19	290	.6138	.6138	.5318	.5318	0.00	0.00
0.2–0.3	27	427	.5457	.5504	.4858	.4936	0.47	0.79
0.3–0.4	28	712	.5351	.5492	.4587	.4720	1.40	1.33
0.4–0.5	15	433	.5266	.5543	.4990	.5284	2.77	2.94
0.5–0.6	9	830	.5482	.6410	.3185	.3663	9.28	4.79

Recorded adaptation protocol. The backbone is Llama 2 7B with the task adapter merged before adding the user adapter. A profile split ratio of 0.5 partitions the initially visible prefix and subsequent stream; it is distinct from the drift-score bin on the horizontal axis. Logs show that the static base receives only its initial warmup training. The early SAF prototype additionally permits drift-triggered updates, using semantic windows of size 8, threshold 0.2, memory size 32, and update segments of at most 16 examples. Its replay budget is 8 examples with semantic-diversity selection; the recorded anchor and distillation weights are 0.05 and 0.2, with distillation temperature 2. The semantic encoder is all-MiniLM-L6-v2, with tag/text weights 3:1. The auxiliary MoE curve uses the saved prototype-routing variant. These settings differ from the final zero-replay, relative-loss SAF controller and are reported only as historical motivation.

What the trend establishes. Both recorded metric gaps strictly increase across the five bins. The result is useful evidence that the relative value of continued adaptation can be greater in groups with higher recorded drift scores. It is a comparison of different historical cohorts, with no uncertainty interval or controlled manipulation of drift. In particular, the highest bin contains nine users and disproportionately many queries. The available historical output metadata does not record a training seed.

Historical bin provenance. We preserve the bin labels attached to the saved results. The original cohort paths recorded in those artifacts are no longer present, and the currently exported drift manifests have different membership. Consequently, original bin assignments cannot be independently reconstructed from the current exports. We do not apply the main experiment's cohort hashes or training-overlap audit to this historical diagnostic. Restoring the original cohort snapshots would be necessary to reproduce its data selection exactly.