SAIL: LEARNING WHAT TO PERSONALIZE FROM SPARSE GUI INTERACTION HISTORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse interaction history is an ambiguous basis for personalization because observed actions may reflect the task, interface, or temporary context rather than transferable user tendencies. We introduce SAIL, a population relative framework for cold start GUI agents that compares historical behavior with plausible alternatives in the same context. SAIL aggregates these deviations into a compact probabilistic user state and applies query conditioned residual corrections to Population predictions. On FingerTip 20K, SAIL improves User Macro Sim1 by 7.48 points on average over the strongest prior history baselines and retains a 1.61 point average gain under a strict control matched for observable support, active capacity, and training budget. It also reaches 43.25% GUI execution success and retains personalization gains on held out users and Persona2Web. Controlled behavioral and adverse history tests show that task relevance alone does not determine history value: transferable user conditioned patterns help, while forced, misleading, or wrong user histories can still cause negative transfer. Overall, population relative modeling provides an effective way to extract transferable user information from sparse GUI interaction history.

1 INTRODUCTION

GUI agents have made substantial progress in understanding interfaces and executing tasks across web, mobile, and desktop environments (Deng et al., 2023; Zhou et al., 2024; Liu et al., 2024; Cheng et al., 2024; Hong et al., 2024; Wang et al., 2024; Drouin et al., 2024; Xie et al., 2024; Rawles et al., 2025; Wu et al., 2025b). Most systems nevertheless assume that the desired task is already specified. A proactive agent faces an earlier problem: inferring what the user wants from the current GUI context before receiving an explicit instruction. Correct execution alone is insufficient when the inferred intent does not match the user's needs.

Interaction history can help resolve this ambiguity by revealing recurring choices and routines. Existing approaches use retrieval, user representations, or parameter-efficient adaptation to incorporate such history (Salemi et al., 2024; Tan et al., 2024a;b), including in personalized web and GUI agents (Cai et al., 2025; Kim et al., 2026; Hu et al., 2026a; Lyu et al., 2026; Yang et al., 2026). Under cold start, however, the agent must infer useful user information from only a few interactions (Lee et al., 2019; Dong et al., 2020; Lin et al., 2021).

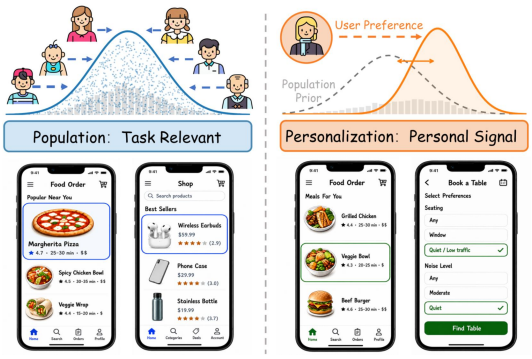


Figure 1: Task-relevant behavior and personal variation under sparse history.

The difficulty is that observed behavior is not a direct description of user preference. An action may reflect a habitual choice, a temporary task requirement, or the options available in the interface. Choosing the only available action says little about preference, whereas a recurring operation across applications may inform future intent.

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.

History can be task-relevant yet provide a misleading basis for user modeling. Using it indiscriminately risks incorrect personalization, while ignoring it discards potentially reusable routines. Figure 1 illustrates this distinction.

We introduce **SAIL** for predicting user intent from the current GUI context and sparse interaction history. Its central idea is to interpret a past choice relative to alternatives that are plausible in the same context, rather than treating the observed action itself as a general preference. A frozen shared predictor, termed the Population model, supplies this reference. SAIL aggregates differences from the reference into a compact user state using learned Gaussian factors, so historical interactions need not contribute equally to every state dimension. The current query then determines how this reusable state adjusts intent generation, separating user-state construction from its application in a new context. Here, attribution concerns predictive differences, not causal identification of preference.

Our contributions are:

- **Context-aware user modeling.** We study the predictive value of historical behavior beyond its original task context in cold-start intent prediction.
- **Population-relative personalization.** SAIL combines an observation-conditioned Population reference, Gaussian user-state aggregation, and query-conditioned residual adaptation.
- **Extensive evaluation.** We assess end-to-end gains, matched-control differences, negative transfer, and generalization to held-out users and Persona2Web.

2 Related Work

Personalized GUI Agents. GUI-agent research has developed grounded interaction and autonomous control across web, mobile, and desktop environments (Deng et al., 2023; Yao et al., 2022; Zhou et al., 2024; Drouin et al., 2024; Xie et al., 2024; Rawles et al., 2025; Wu et al., 2025b; Xu et al., 2025). Personalized agents extend this setting by using user history to resolve implicit intent and adapt interaction (Cai et al., 2025; Hu et al., 2026a; Kim et al., 2026; Lyu et al., 2026; Yang et al., 2026). SAIL focuses on the intent-prediction stage: given the current GUI context and limited history, it models how past choices should affect the predicted user intent.

Cold-Start User Modeling. Cold-start methods infer user representations from limited observations through meta-learning, memory-based adaptation, or neural processes (Lee et al., 2019; Dong et al., 2020; Lin et al., 2021). LLM personalization also uses retrieval, learned user representations, and parameter-efficient adaptation (Salemi et al., 2024; Tan et al., 2024a;b; Zollo et al., 2025). Conditional and Attentive Neural Processes, together with Bayesian context aggregation, provide related approaches to summarizing sparse observations (Garnelo et al., 2018; Kim et al., 2019; Volpp et al., 2021). SAIL uses learned Gaussian factors to aggregate user history, while representing each interaction relative to a context-matched reference.

Selective Personalization. Long-term memory systems store and retrieve accumulated interaction history (Wang et al., 2023; Packer et al., 2023; Zhong et al., 2024; Wu et al., 2025a). Recent work further studies when personal information should influence a response, including over-personalization, context-dependent preference selection, and utility-aware memory admission (Hu et al., 2026b; Yoon et al., 2026; Wang et al., 2026a). SAIL shares the goal of avoiding inappropriate history-conditioned changes, while focusing on sparse GUI behavior whose implications for user modeling are uncertain. It combines context-relative history representation with query-conditioned adaptation, and evaluates both predictive gains and negative transfer.

3 Method

3.1 Problem Formulation and Overview

Given the current GUI context and a small history of completed interactions, our goal is to predict the user's intent before an explicit instruction is provided. Let u denote the user, x_q the current multimodal GUI context, and $y_q = (y_{q,1}, \ldots, y_{q,T})$ the target intent sequence. The available support

history and prediction model are

$$\mathcal{H}_u^S = \{(x_i, y_i)\}_{i=1}^S, \quad S \le 20, \qquad p_\theta(y_q \mid x_q, \mathcal{H}_u^S). \tag{1}$$

Cold start refers here to limited user-specific history at inference time, not necessarily to users absent from training; fully unseen users are evaluated separately in Sec. 4.4. We denote the frozen Population generator by $p_0(y \mid x)$ and its support and query context representations by h_i and h_q.

A history item records what the user did in a particular situation. Treating a task-driven action as a general preference can therefore produce an incorrect user model. SAIL compares each historical intent with a reference constructed for its original context, summarizes the differences in a latent user state, and conditions the resulting adjustment on the current query. The Population backbone remains frozen; only the user-state inference and residual-adaptation pathways are trained.

3.2 Population-Relative Support Representation

For each historical context x_i, we sample M plausible intents from the Population model:

$$\mathcal{C}_i = \{c_{im}\}_{m=1}^M, \quad c_{im} \sim p_0(\cdot \mid x_i), \qquad \ell_{im} = \log p_0(c_{im} \mid x_i). \tag{2}$$

Let $v_{im} \in \mathbb{R}^H$ and $o_i \in \mathbb{R}^H$ denote frozen representations of candidate c_{im} and observed intent y_i. A single GUI context may admit several reasonable intents. We therefore construct a reference by weighting the candidates according to both their Population likelihood and their similarity to the observed historical intent:

$$s_{im} = \frac{\ell_{im}}{\tau_p} - \frac{1 - \cos(v_{im}, o_i)}{\tau_c}, \qquad \rho_{im} = \frac{\exp(s_{im})}{\sum_{m'} \exp(s_{im'})}, \qquad r_i = \sum_{m=1}^M \rho_{im} v_{im}. \tag{3}$$

The reference r_i combines model plausibility with compatibility with the observed historical interaction. It is conditioned on o_i, rather than being an unconditional population mean. This construction uses only the observed historical target y_i; the query target y_q is never accessed.

We represent the difference between the observed intent and this reference as

$$d_i = o_i - \mathrm{sg}(r_i), \qquad \bar{d}_i = \frac{d_i}{\max(\|d_i\|_2, \epsilon)}, \qquad \delta_i = \|d_i\|_2 f_\delta(\bar{d}_i), \qquad \delta_i \in \mathbb{R}^{D_z}, \tag{4}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient. The residual magnitude records the size of this difference, while f_δ maps its direction into the latent user space. The resulting δ_i is an input to user modeling, not a direct measurement of preference: it may also contain effects of context constraints or errors in the shared predictor.

An interaction need not be equally informative about every part of the user state. We therefore predict a dimension-wise precision λ_i and observability vector a_i:

$$\lambda_i = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min})\sigma(f_\lambda([h_i; o_i; r_i; |d_i|])), \qquad a_i = \sigma(f_a(h_i)). \tag{5}$$

In the factor model below, $a_{i,j}$ links user-state coordinate $z_{u,j}$ to the observed residual $\delta_{i,j}$, while $\lambda_{i,j}$ controls the penalty for a mismatch. Both are learned from the support interaction; precision is not a calibrated score of that interaction's correctness.

3.3 User-State Inference with Learned Gaussian Factors

Let $z_u \in \mathbb{R}^{D_z}$ be a latent user state shared across the support interactions, with $D_z = 256$ in our implementation (Appendix B.2). Each interaction defines a Gaussian-shaped factor that favors states whose scaled coordinates $a_i \odot z_u$ match the observed residual δ_i:

$$\psi_\phi(z_u; x_i, y_i) = \exp\left[-\frac{1}{2}\sum_{j=1}^{D_z} \lambda_{i,j}(\delta_{i,j} - a_{i,j} z_{u,j})^2\right], \tag{6}$$
$$p(z_u) = \mathcal{N}(0, \lambda_0^{-1} I), \qquad \lambda_0 = 1.$$

The zero-centered prior represents the absence of a systematic deviation from Population behavior. Combining the support factors yields

$$q_\phi(z_u \mid \mathcal{H}_u^S) = \frac{p(z_u)\prod_{i=1}^S \psi_\phi(z_u; x_i, y_i)}{Z_\phi(\mathcal{H}_u^S)}.$$

Because the factors are quadratic, aggregation is closed-form:

$$\Lambda_{u,j} = \lambda_0 + \sum_{i=1}^{S} \lambda_{i,j} a_{i,j}^2, \qquad \mu_{u,j} = \Lambda_{u,j}^{-1} \sum_{i=1}^{S} \lambda_{i,j} a_{i,j} \delta_{i,j}, \qquad \Sigma_{u,j} = \Lambda_{u,j}^{-1}. \tag{7}$$

Hence,

$$q_\phi(z_u \mid \mathcal{H}_u^S) = \mathcal{N}(\mu_u, \mathrm{diag}(\Sigma_u)). \tag{8}$$

The mean μ_u summarizes the inferred behavioral deviation, while Σ_u records the variance assigned to each coordinate by the factor model.

During training, we sample $z_u = \mu_u + \Sigma_u^{1/2} \odot \varepsilon$, $\varepsilon \sim \mathcal{N}(0, I)$; at inference, we use $z_u = \mu_u$. The posterior depends on history only through the additive sufficient statistics

$$\sum_i \lambda_i \odot a_i^{\odot 2}, \qquad \sum_i \lambda_i \odot a_i \odot \delta_i, \tag{9}$$

For fixed network parameters and cached per-interaction factors, a new interaction can be incorporated by updating these two sums without re-encoding the existing history.

We summarize the reduction in modeled variance by

$$I_u = \frac{1}{2D_z} \sum_{j=1}^{D_z} \log \frac{\Lambda_{u,j}}{\lambda_0}. \tag{10}$$

A larger I_u means that the model assigns a narrower distribution to the user state than the prior. It does not by itself establish that the user model is more accurate.

3.4 Posterior-Gated Residual Adaptation

A behavior learned from history need not apply to every new situation. We therefore condition the user representation on the current query:

$$m_u = f_m([z_u; \log \Sigma_u]), \qquad g_q = \sigma(W_g[m_u; h_q] + b_g), \qquad \widetilde{m}_{u,q} = g_q \odot m_u. \tag{11}$$

The projector f_m receives both the inferred state and its modeled variance. The gate g_q then reweights the resulting representation according to the current GUI context.

We use posterior concentration to set an instance-dependent global adjustment strength:

$$\alpha_u = (1 - \exp(-\max(I_u, 0)))\, \mathbf{1}[S_u > 0]. \tag{12}$$

The two gates have different roles: g_q selects how the user representation is applied in the current context, while α_u scales the overall adjustment. Here S_u is the number of valid support interactions, so $\alpha_u = 0$ when no history is available.

SAIL adds a low-rank residual without updating the Population decoder. Let $h_{q,t}^{(0)}$ be its frozen hidden state at decoding step t; h_q is the pre-response state used by g_q. Then

$$q_t = \tanh(W_q h_{q,t}^{(0)} + b_q), \qquad u_q = \tanh(W_m \widetilde{m}_{u,q} + b_m), \tag{13}$$

$$p_t = q_t \odot u_q, \qquad b_t = \sigma(W_b[p_t; q_t; u_q] + b_b)\,, \tag{14}$$

$$\Delta h_t = W_o(b_t \odot p_t) + b_o. \tag{15}$$

This interaction allows the same user state to produce different corrections across decoding steps.

Using the detached frozen vocabulary projection $E_0 \in \mathbb{R}^{|V| \times H}$,

$$\Delta \ell_t = \frac{\Delta h_t E_0^\top}{\sqrt{H}}, \qquad \ell_t = \ell_t^{(0)} + \mathrm{softplus}(\gamma)\alpha_u\ \mathrm{RMSCalibrate}(\Delta \ell_t). \tag{16}$$

Here $\mathrm{RMSCalibrate}$ centers the residual logits and rescales them to unit target RMS with numerical floor 10^{-6}. A small unnormalized residual therefore does not by itself ensure a small final correction: global strength depends on α_u and the learned scale $\mathrm{softplus}(\gamma)$. Appendix B.5 discusses this distinction. Candidate intents are used only to construct historical references; inference still generates an intent rather than selecting from that candidate set.

3.5 Training and Model Behavior

Learning objective. SAIL is optimized with three complementary objectives:

$$\mathcal{L} = \lambda_q \mathcal{L}_{\text{query}} + \lambda_e \mathcal{L}_{\text{support}} + \lambda_d \mathcal{L}_{\text{dir}}. \tag{17}$$

They supervise target prediction, latent support modeling, and residual correction, respectively.

The query objective is the teacher-forced negative log-likelihood:

$$\mathcal{L}_{\text{query}} = -\sum_t \log p_\theta \left(y_{q,t} \mid y_{q,<t}, x_q, \mathcal{H}_u^S \right). \tag{18}$$

The support objective trains the latent observation factors:

$$\mathcal{L}_{\text{support}} = \frac{1}{2SD_z} \sum_{i,j} \left[\lambda_{i,j} \left(\delta_{i,j} - a_{i,j} z_{u,j} \right)^2 - \log \lambda_{i,j} \right]. \tag{19}$$

This objective encourages the inferred state to fit the residual observations in the support history. For $S = 0$, we set $\mathcal{L}_{\text{support}} = 0$.

Finally, $\mathcal{L}_{\text{dir}}$ corrects supervised positions where the frozen Population model predicts an incorrect top-1 token:

$$\begin{aligned} \hat{y}_t^{\text{Pop}} &= \arg\max_v \text{stopgrad}\left(\ell_{t,v}^{\text{Pop}} \right), \qquad \mathcal{A} = \{ t : \hat{y}_t^{\text{Pop}} \neq y_t \}, \\ \mathcal{L}_{\text{dir}} &= \frac{1}{|\mathcal{A}|} \sum_{t \in \mathcal{A}} \max\left(0,\ 0.5 - \left[\ell_{t,y_t}^{\text{SAIL}} - \ell_{t,\hat{y}_t^{\text{Pop}}}^{\text{SAIL}} \right] \right). \end{aligned} \tag{20}$$

The loss is zero when $\mathcal{A}$ is empty. We use

$$(\lambda_q, \lambda_e, \lambda_d) = (1.0,\ 0.1,\ 1.0).$$

No-history behavior and incremental updates. For $S = 0$, $q_\phi(z_u \mid \mathcal{H}_u^0) = p(z_u)$ and $I_u = 0$, hence $\alpha_u = 0$. Therefore,

$$p_\theta(y \mid x_q, \mathcal{H}_u^0) = p_0(y \mid x_q).$$

SAIL thus exactly recovers the Population predictor when no user history is available.

With nonempty history, the additive statistics in Eq. (9) allow incremental state updates. However, repeated or conflicting interactions can still reduce the modeled variance without improving the user model. Appendix B.5 derives the update properties and explains why concentration alone does not guarantee a vanishing correction for uninformative history. The support-objective ablations and history stress tests are reported in Appendices C.7 and C.9.

4 Experiments

We evaluate SAIL on FingerTip-20K and Persona2Web along four questions: **RQ1.** Does SAIL improve intent prediction under sparse history and matched controls? **RQ2.** How do user rules, environment changes, and shared Population error affect the value of history? **RQ3.** Which components govern personalization gain and negative transfer? **RQ4.** Do the gains transfer to unseen users and a new domain?

4.1 Experimental Setup

Benchmarks and protocols. FingerTip-20K contains 21,437 episodes from 95 users across 506 applications (Yang et al., 2026). We evaluate 1,633 queries from 83 eligible users chronologically at $S \in \{0, 1, 2, 4, 8, 20\}$, using the most recent valid interactions preceding each query. GUI execution uses $S = 4$; unseen-user and Persona2Web transfer are evaluated in Sec. 4.4. Split construction and leakage audits are in Appendices A.1 and A.2.

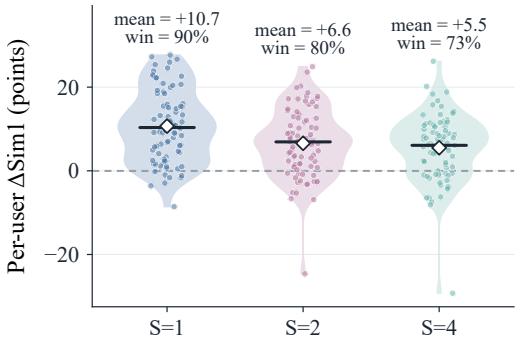


Figure 2: User-level gains over Recent-S. Per-user mean Sim1 differences at $S = 1, 2, 4$; annotations report the mean gain and fraction of users with a positive gain.

Baselines and controls. Table 1 includes generalist and GUI-agent references (OpenAI, 2025; GLM-V Team, 2025; Bai et al., 2025b; Qwen Team, 2025; Bai et al., 2025a; Ye et al., 2025; Qin et al., 2025). History baselines cover direct conditioning through Recent-S and Similarity-Ordered, retrieval through Retrieve (Lyu et al., 2026), memory through MemoryBank (Zhong et al., 2024) and Me-Agent HPM (Wang et al., 2026b), and user modeling through LLM-UM (Wang et al., 2025) and HIM-Agent (Lyu et al., 2026). Learned controls compare adaptation mechanisms: Query-Only Adapter uses no history, Learned-History predicts from query and history, Deterministic Residual applies a deterministic history-conditioned correction, and Per-User LoRA (Hu et al., 2022) adapts user-specific parameters. MHR removes the Population reference while matching multimodal support, registered architecture, and training budget; MHR-ActiveCap additionally matches active parameter count. Definitions and matched-control details are in Appendices A.1 and C.7; training details are in Appendix B.3.

Metrics and evaluation. Following FingerTip-20K, $\mathrm{Sim}_1 = (S_1 + S_2)/2$ combines MiniLM cosine and normalized Levenshtein similarity. **User-Macro Sim1** averages queries within users, then weights users equally; scores are scaled by 100. Execution reports SR_2 and trajectory personalization Sim_2; selective diagnostics report gain, harm, and negative-transfer rate at matched coverage. Table 1 reports mean±std over three independent runs. Trainable methods are independently retrained, whereas non-trainable baselines use three independent inference runs. Mechanism intervals use paired user-level bootstrap with Holm correction, not run-level significance tests. Exact Population-recovery methods have a deterministic $S = 0$ score. Metric definitions, execution-aggregation checks, and format/slot diagnostics are in Appendices A.1 and C.2.

4.2 Main Results: Cold-Start Personalization—RQ1

Cold-Start Performance. At the system level, SAIL gains **7.48** points on average over the strongest prior history-based baselines at $S = 1, 2, 4$. We separately test the Population-relative construction with MHR-ActiveCap, which matches observable support, active parameter count, and training budget; SAIL retains $+2.27/+1.44/+1.13$ points at $S = 1/2/4$, averaging $+1.61$ (Table 3). The former is an end-to-end comparison, whereas the latter is the stricter matched comparison. One interaction recovers 88.7% of SAIL's $S = 20$ improvement over Population. Direct conditioning becomes more competitive with additional history, so SAIL's main advantage is using very few interactions effectively.

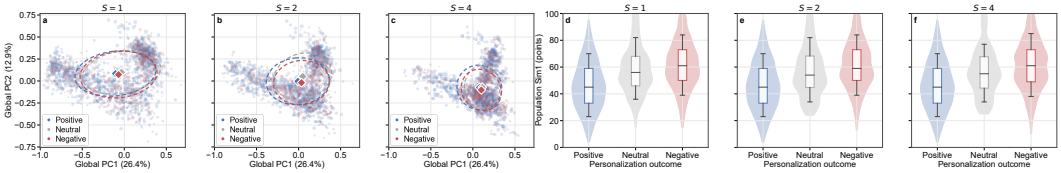


Figure 3: User-state geometry and baseline prediction quality. Panels a–c project user states onto shared principal components at $S = 1, 2, 4$; d–f show the corresponding Population scores. Positive/neutral/negative groups use post-hoc Sim1 gains over Population: > 1, $[-1, 1]$, and < -1 points, respectively.

Consistency Across Queries and Users. Figures 2 and 4 show broad low-shot gains across users and queries, with a shrinking advantage as support grows. Negative tails remain: average improvement does not imply that every query benefits.

Where Personalization Helps. In Figure 3, gains tend to accompany weaker Population predictions, whereas harmful corrections more often disturb already relevant intents. This outcome-

Table 1: Intent prediction on FingerTip-20K. User-Macro Sim1 scaled by 100. Mean±std entries summarize three independent runs: trainable methods are independently retrained, while non-trainable baselines use three independent inference runs. Exact Population-recovery methods share its deterministic $S = 0$ score. A dash at $S = 0$ denotes a history-dependent method without a separately defined no-history configuration. All methods follow the chronological-support protocol; input details are in Appendix A.1.

Method	$S = 0$	$S = 1$	$S = 2$	$S = 4$	$S = 8$	$S = 20$
General GUI Models						
GPT-5.1	56.20±0.32	57.10±0.35	58.20±0.34	59.05±0.32	60.10±0.30	61.35±0.28
GLM-4.5V	54.90±0.35	55.65±0.38	56.80±0.36	57.70±0.34	58.95±0.31	60.40±0.30
QwenVL-Max	55.60±0.33	56.45±0.36	57.55±0.34	58.60±0.32	59.85±0.30	61.25±0.28
Qwen-QVQ-Max	**57.00±0.31**	57.80±0.34	58.70±0.32	59.65±0.30	60.90±0.29	62.00±0.27
Qwen3-VL-8B	54.75±0.34	55.80±0.37	57.10±0.35	58.30±0.33	59.55±0.31	61.00±0.29
GUI-Owl-7B	53.90±0.36	54.95±0.39	56.15±0.37	57.40±0.35	58.80±0.32	60.20±0.30
UI-TARS-1.5-7B	55.10±0.34	56.20±0.37	57.55±0.35	58.90±0.33	60.05±0.30	61.50±0.28
Personalization Methods						
Population	50.61	–	–	–	–	–
Recent-S	–	51.67±0.24	55.48±0.31	56.91±0.27	60.72±0.29	63.70±0.25
Similarity-Ordered	–	51.62±0.14	52.37±0.28	56.03±0.32	59.29±0.30	62.22±0.27
MemoryBank	–	51.02±0.33	55.33±0.39	57.07±0.35	60.21±0.32	63.70±0.30
Me-Agent HPM	–	51.33±0.35	55.07±0.41	56.94±0.38	60.43±0.36	63.63±0.31
Retrieve	–	51.67±0.36	55.37±0.40	57.23±0.37	60.29±0.34	63.22±0.32
LLM-UM	–	50.53±0.48	51.77±0.52	53.32±0.47	55.40±0.45	54.39±0.43
HIM-Agent	–	51.55±0.42	55.23±0.46	55.49±0.44	55.40±0.40	60.23±0.37
Task-Specific Adaptation Methods						
Query-Only Adapter	55.84±0.31	55.84±0.31	55.84±0.31	55.84±0.31	55.84±0.31	55.84±0.31
Learned-History	–	58.31±0.54	60.14±0.47	61.46±0.41	62.09±0.38	62.88±0.35
Deterministic Residual	50.61	60.21±0.48	60.83±0.43	61.92±0.38	62.20±0.35	62.45±0.33
Per-User LoRA	–	59.35±0.52	60.42±0.47	61.31±0.40	–	–
Matched History Residual	50.61	59.92±0.46	60.51±0.42	61.18±0.36	62.25±0.34	63.08±0.32
SAIL	50.61	**62.35±0.42**	**62.05±0.36**	**62.43±0.33**	**62.57±0.29**	**63.85±0.31**

conditioned description motivates examining query-specific application separately from user-state construction; it is not an inference-time criterion for selecting beneficial queries.

Downstream GUI Execution. With a fixed executor on 400 paired tasks, SAIL reaches 43.25% success versus 40.75% for Retrieve and 42.00% for MHR, with higher trajectory alignment and similar latency (Table 2). The executor receives only the predicted intent, not the reference intent or SAIL's state; thus the comparison tests the effect of intent specification. Appendix C.3 details aggregation and paired resampling.

Table 2: Representative GUI execution results. History-aware methods use $S = 4$; the extended comparison is in Appendix C.3.

Method	SR_2 ↑	Sim_2 ↑	Step↓	Time↓	Tok.↓
Population	38.50	1.05	1.23	10.18	2456
Recent-S	39.50	1.10	1.23	10.31	2615
MemoryBank	40.25	1.13	1.22	10.42	2655
Retrieve	40.75	1.15	1.21	10.47	2632
MHR	42.00	1.20	1.20	10.50	2528
SAIL	**43.25**	**1.25**	**1.18**	10.53	2510

Beyond Surface-Form Improvement. Format-normalized and slot-level evaluation retains SAIL's advantage, supporting better task specification beyond output formatting (Appendix C.2).

4.3 User Modeling and Mechanism Analysis—RQ2, RQ3

User Modeling Across Contexts. We vary user choice rules and GUI layouts in 320 controlled queries. SAIL gains 7.6 Sim1 points over Population when users follow different rules in the same interface, and 5.5 when the rule is held fixed across interface changes. For forced actions, its mean gain is 0.0; for shared errors, Query-Only Adapter improves more than SAIL, by 6.1 versus 0.6 points. These results distinguish reusable user behavior from generic error correction. Appendix C.8 also compares selection scores at 50% coverage, first across score–prediction configurations and then with SAIL predictions held fixed.

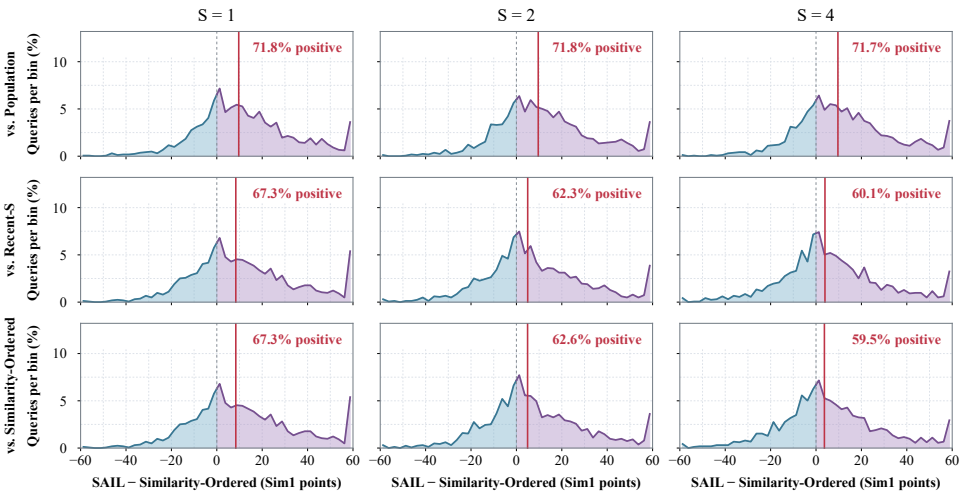


Figure 4: Query-level gains under limited history. SAIL-minus-reference Sim1 distributions at $S \in \{1, 2, 4\}$; positive fractions exclude ties.

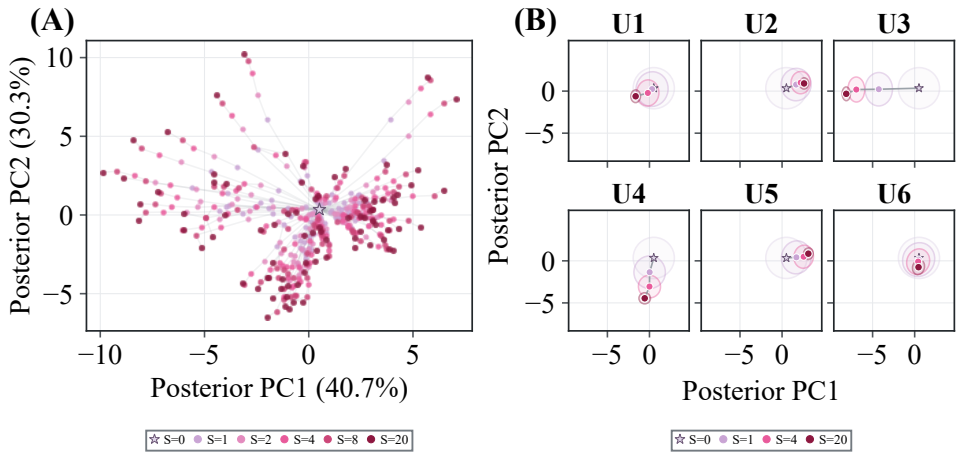


Figure 5: Posterior geometry across support budgets. Panel A shows departure from the prior; Panel B shows model-implied contraction, not calibrated evidence reliability.

What the Controls Show. Replacing a user's history with another user's interactions from similar tasks lowers Sim1 by 3.91–4.86 points at $S = 1, 2, 4$: task similarity alone does not preserve the information in the user's own choices. We then remove subtraction with and without allowing the history-weighting network to use the Population reference. The one-shot drops are 6.80 and 1.60 points, respectively: the benefit of subtraction depends on how history is weighted. Matching the scale of raw observations still leaves an advantage for Population-relative representations. Averaging residuals, omitting variance from the state input, or using only support count for global strength each lowers accuracy (Table 3). Removing the dimension gate is less conclusive because its interval includes zero. Appendix C.7 gives the controls.

Robustness to History and Candidate Sampling. At $S = 4$, we test Population-typical, repeated, conflicting, and task-matched wrong-user histories, measuring Sim1, negative-transfer rate (NTR; a drop of more than five points from Population), and token-level Jensen–Shannon prediction shift. With the inferred state and residual head fixed, adaptive rather than fixed strength lowers NTR by 8.7–15.4 percentage points across these conditions. For wrong-user history, NTR falls from 53.1% to 39.4% and prediction shift from 0.082 to 0.034; harmful corrections remain. On unmodified queries, NTR falls from 18.1% to 14.6%. Separately, five candidate-sampling seeds at the SAIL checkpoint used for this diagnostic give Sim1 standard deviations of 0.11/0.09/0.07 points at $S = 1, 2, 4$ and posterior cosine similarity of 0.972–0.984 (Figure 6). This tests sampling variability, not retraining variability (Appendices C.9 and C.7).

Table 3: Matched and mechanism controls. User-Macro Sim1; Δ@1 is SAIL minus control at $S = 1$. MHR-ActiveCap is the strict control matching observable support, active parameter count, and training budget. For this row we report the three-run mean difference; training-run dispersion for both methods is reported separately in Appendix C.7, and no run-level significance test is claimed. Other rows report paired user-bootstrap intervals and Holm-adjusted p-values.

Method	S=1	S=2	S=4	Δ@1	**95% CI**	p_{Holm}
MHR-ActiveCap	60.08	60.61	61.30	+2.27	–	–
Mean Aggregation Only	60.88	61.19	62.12	+1.47	[+0.82,+2.11]	0.002
Fixed Variance Input	61.72	61.63	62.26	+0.63	[+0.12,+1.14]	0.031
Count Gate Only	61.11	61.52	62.10	+1.24	[+0.64,+1.84]	0.004
w/o Dim. Gate	60.90	62.00	62.21	+1.45	[−0.10,+3.00]	0.084
SAIL	**62.35**	**62.05**	**62.43**	–	–	–

Table 4: Unseen-user and Persona2Web generalization. FingerTip-20K reports User-Macro Sim1; Persona2Web uses $S = 4$.

	User-disjoint FingerTip-20K			**Persona2Web,** $S = 4$			
Method	$S = 1$	$S = 2$	$S = 4$	P_{web}	P_{pref}	Intent	Success
Recent-S	49.16	51.38	53.97	34.4	25.0	48.4	2.0
Learned-History	53.84	55.29	56.43	41.2	33.5	50.4	4.0
Deterministic Residual	54.92	56.08	57.01	44.0	36.5	50.7	6.0
Matched History Residual	55.27	56.33	57.21	44.9	37.4	50.8	6.0
MHR-ActiveCap	55.34	56.39	57.28	–	–	–	–
SAIL	**56.47**	**57.38**	**58.16**	**46.4**	**39.0**	**51.2**	**8.0**

4.4 Generalization to Unseen Users and New Domains—RQ4

On 260 queries from 13 users excluded from all model training, SAIL exceeds MHR-ActiveCap by 1.13/0.99/0.88 points at $S = 1, 2, 4$. Without retraining on Persona2Web, it improves P_{web} and P_{pref} over MHR by 1.5 and 1.6 points. Task success rises from 6.0% to 8.0%, but the 95% interval for this difference is $[-1.8, +5.8]$ percentage points. Personalization thus transfers to web tasks; the success-rate improvement remains uncertain with only 50 personas. Appendix C.4 gives the held-out-user protocol and transfer intervals.

Efficiency. In the $S = 1$ validation test, increasing M from 8 to 16 changes Sim1 from 62.35 to 62.41, but raises support preparation from 718 to 1,395 ms; cached query latency changes only from 413 to 414 ms. Our setting is $M = 8$. Across $S = 1$–20, SAIL keeps online input at about 2,335 tokens, whereas Recent-S grows from 2,371 to 3,073. At $S = 20$, this saves 737 tokens, or 24.0%, with scores of 63.85 versus 63.70. These savings concern online generation: new interactions still require support processing, and checkpoint changes require recomputing cached factors (Appendix D.1).

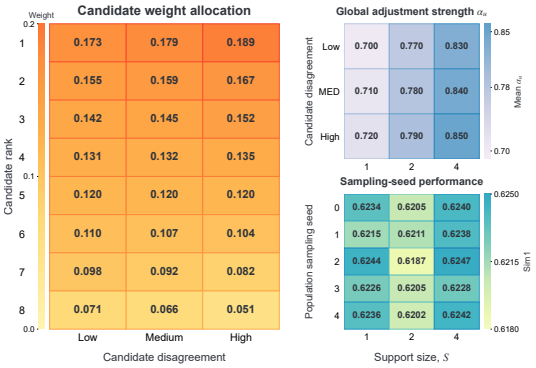


Figure 6: Population sampling analysis.

5 Conclusion

SAIL combines context-matched references, Gaussian aggregation, and query-conditioned corrections for cold-start GUI intent prediction. It improves over prior history baselines and retains positive gains under input-, active-capacity-, and training-budget-matched controls. History substitutions and controlled tests further show that a user's own interaction choices provide predictive information beyond task similarity and shared prediction error, with personalization gains transferring to unseen users and web tasks. Overall, population-relative modeling helps extract transferable user-conditioned information from sparse histories for cold-start intent prediction.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, improving clarity, and manuscript organization. All AI-assisted text was carefully reviewed and revised by the authors. The authors take full responsibility for the accuracy, scientific content, and final presentation of the paper.

ETHICS STATEMENT

This work uses existing research benchmarks and does not collect new human-subject data. Personalized GUI agents may raise privacy concerns and may produce unsupported personalization when sparse interaction histories are misinterpreted. We therefore audit temporal and episode-level leakage and evaluate negative transfer under wrong-user and ambiguous histories. SAIL's learned residuals and posterior quantities are modeling signals and should not be interpreted as calibrated evidence of users' true preferences. Real-world deployment should include appropriate privacy safeguards, user control, and oversight.

REPRODUCIBILITY STATEMENT

We provide model formulations, training and evaluation details, and additional analyses in the main paper and appendices. A reference implementation of the core SAIL modules is available in our anonymous repository: https://anonymous.4open.science/r/SAIL-7E9B.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Hongru Cai, Yongqi Li, Wenjie Wang, Fengbin Zhu, Xiaoyu Shen, Wenjie Li, and Tat-Seng Chua. Large language models empowered personalized web agents. In *Proceedings of the ACM Web Conference (WWW)*, 2025.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Manqing Dong, Feng Yuan, Lina Yao, Xiwei Xu, and Liming Zhu. Mamo: Memory-augmented meta-optimization for cold-start recommendation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2020.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. Workarena: How capable are web agents at solving common knowledge work tasks? In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo J. Rezende, and S. M. Ali Eslami. Conditional neural processes. In *Proceedings of the 35th International Conference on Machine Learning(ICML)*, 2018.

GLM-V Team. Glm-4.1v-thinking and glm-4.5v: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Die Hu, Jingguo Ge, Weitao Tang, He Kong, Liangxiong Li, and Bingzhen Wu. Orion: Steering personalized web agents via global-micro profiling and adaptive intent tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2026a.

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 *Proceedings of the International Conference on Learning Representations(ICLR)*, 2022.

Yulin Hu, Zimo Long, Jiahe Guo, Xingyu Sui, Xing Fu, Weixiang Zhao, Yanyan Zhao, and Bing Qin. OP-Bench: Benchmarking over-personalization for memory-augmented personalized conversational agents. *arXiv preprint arXiv:2601.13722*, 2026b.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. In *Proceedings of the International Conference on Learning Representations(ICLR)*, 2019.

Serin Kim, Sangam Lee, and Dongha Lee. Persona2web: Benchmarking personalized web agents for contextual reasoning with user history. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026.

Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. Melu: Meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2019.

Xixun Lin, Jia Wu, Chuan Zhou, Shirui Pan, Yanan Cao, and Bin Wang. Task-adaptive neural process for user cold-start recommendation. In *Proceedings of the ACM Web Conference (WWW)*, 2021.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

Yibo Lyu, Gongwei Chen, Rui Shao, Weili Guan, and Liqiang Nie. Personalalign: Hierarchical implicit intent alignment for personalized gui agent with long-term user-centric records. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026.

OpenAI. Gpt-5.1 instant and gpt-5.1 thinking system card addendum. OpenAI System Card, 2025.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Yujia Qin, Yining Ye, Junjie Fang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025.

Qwen Team. Qvq-max: Think with evidence. Qwen Technical Blog, 2025.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William E. Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Kenji Toyama, Robert James Berry, Divya Tyamagundlu, Timothy P. Lillicrap, and Oriana Riva. Androidworld: A dynamic benchmarking environment for autonomous agents. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. Lamp: When large language models meet personalization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Zhaoxuan Tan, Zheyuan Liu, and Meng Jiang. Personalized pieces: Efficient personalized large language models through collaborative efforts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024a.

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 the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024b.

Michael Volpp, Fabian Flürenbrock, Lukas Großberger, Christian Daniel, and Gerhard Neumann. Bayesian context aggregation for neural processes. In *Proceedings of the International Conference on Learning Representations(ICLR)*, 2021.

Jing Wang, Zhu Wang, Yifan Guo, Yulong Yang, and Yunji Liang. TRACE-Memory: Public-conditioned retrieval and utility-aware evidence admission for personalized generation. *arXiv preprint arXiv:2608.08446*, 2026a.

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *arXiv preprint arXiv:2401.16158*, 2024.

Lu Wang, Di Zhang, Fangkai Yang, Pu Zhao, Jianfeng Liu, Yuefeng Zhan, Hao Sun, Qingwei Lin, Weiwei Deng, Dongmei Zhang, et al. Lettingo: Explore user profile generation for recommendation system. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2985–2995, 2025.

Shuoxin Wang, Chang Liu, Gowen Loo, Lifan Zheng, Kaiwen Wei, Huanqian Yan, Xinyi Zeng, Jingyuan Zhang, and Yu Tian. Me-agent: A personalized mobile agent with two-level user habit learning for enhanced interaction. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 24206–24222, 2026b.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025a.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. Os-atlas: Foundation action model for generalist gui agents. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025b.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. Aguvis: Unified pure vision agents for autonomous gui interaction. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

Qinglong Yang, Haoming Li, Haotian Zhao, Xiaokai Yan, Jingtao Ding, Fengli Xu, and Yong Li. Fingertip 20k: A benchmark for proactive and personalized mobile llm agents. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Jiabo Ye, Xi Zhang, Haiyang Xu, et al. Mobile-agent-v3: Foundamental agents for gui automation. *arXiv preprint arXiv:2508.15144*, 2025.

Sangyeon Yoon, Sunkyoung Kim, Hyesoo Hong, Wonje Jeung, Yongil Kim, Wooseok Seo, Heuiyeen Yeen, and Albert No. BenchPreS: A benchmark for context-aware personalized preference selectivity of persistent-memory LLMs. *arXiv preprint arXiv:2603.16557*, 2026.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

Thomas P. Zollo, Andrew Wei Tung Siah, Naimeng Ye, Ang Li, and Hongseok Namkoong. Personalllm: Tailoring llms to individual preferences. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

APPENDIX

A Experimental Protocol and Data Integrity **15**

A.1 Evaluation Protocol, Baselines, and Metrics 15

A.2 Data Provenance and Leakage Audit 17

B Model Implementation and Optimization **18**

B.1 Population Prior: Training Data and Optimization 18

B.2 Implementation Details of SAIL 20

B.3 Training and Optimization Details 21

B.4 Population-Anchored Personalization with a Model-Implied User Residual 24

B.5 Incremental Updates and Interpretation of Posterior Concentration 24

C Extended Quantitative Results **26**

C.1 Full Metric Breakdown 26

C.2 Format-Normalized and Slot-Level Evaluation 26

C.3 Task-Matched Replacement and Extended GUI Execution Results 27

C.4 Full User-Disjoint and Cross-Benchmark Results 28

C.5 Broader Baseline Behavior 29

C.6 User- and Query-Level Heterogeneity 29

C.7 Mechanism Diagnostics Under Limited History 29

C.8 Controlled Behavioral Attribution and Gain–Harm Evaluation 36

C.9 Posterior-Concentration Stress Tests 39

D Efficiency and Execution Diagnostics **39**

D.1 Online Context Accounting 39

D.2 Offline Action-Level Personalization Diagnostics 40

E Qualitative Cases and Supplementary Discussion **41**

A Experimental Protocol and Data Integrity

A.1 Evaluation Protocol, Baselines, and Metrics

FingerTip-20K protocol. FingerTip-20K contains 21,437 mobile interaction episodes from 95 users across 506 applications. We retain 20,000 episodes from 83 users; 12 users and 1,437 episodes are excluded because they do not provide sufficient valid history for the required support budgets. We train SAIL on 14,284 episodes and evaluate it on 1,633 queries from this eligible cohort. Each evaluation query has enough valid earlier interactions for every reported budget. Its support $\mathcal{H}_u^S$ contains the S most recent valid interactions strictly preceding the query, ordered chronologically within the selected window. The windows are nested:

$$\mathcal{H}_u^1 \subset \mathcal{H}_u^2 \subset \mathcal{H}_u^4 \subset \mathcal{H}_u^8 \subset \mathcal{H}_u^{20},$$

with $S \in \{0, 1, 2, 4, 8, 20\}$. The query target and later interactions are excluded. The primary evaluation protocol is temporal within users. A separate $60/10/13$ user split is used only for Population-prior development; after the 1,633-query primary evaluation set is fixed, the same episode-level and query-relative temporal boundary is enforced for Population training, Population selection, and SAIL training, as audited in Appendix A.2. History-aware GUI execution comparisons use $S = 4$.

Baselines. We compare direct-history conditioning, memory-based personalization, and learned adaptation. The following descriptions summarize how each baseline uses user information; the matched residual controls are detailed below.

General GUI models. At $S > 0$, each model receives the same evaluation query and the same frozen $\mathcal{H}_u^S$ support IDs. The current GUI observation is kept unchanged, and each historical interaction is serialized in chronological order using the available pre-target time, scenario, application metadata, and observed intent. The prompt template, model/API version, truncation rule, maximum output length, and decoding hyperparameters are fixed across repetitions. We report three independent API/inference runs for these non-trainable models; only inference-time stochasticity varies across runs. At $S = 0$, the history is empty. These are end-to-end comparisons rather than the matched-input controls below.

Population. The shared predictor generates an intent from the current query inputs without access to interaction history. It provides the zero-history reference for history-conditioned prediction.

Recent-S and Similarity-Ordered. Both methods place historical interactions directly in the prediction context. Recent-S uses the most recent valid interactions in chronological order. Similarity-Ordered uses the same fixed support set but reranks those records by query similarity before serialization. It therefore changes the ordering of a fixed support set rather than retrieving a different set of records. They provide direct-history comparisons without a separate learned user state.

MemoryBank and Me-Agent HPM. These methods use memory-based personalization. MemoryBank draws on stored user interactions, while Me-Agent HPM uses a habit-based memory of user behavior. They compare memory-based use of history with SAIL's population-relative user-state representation.

Retrieve. This retrieval-based baseline independently selects the most relevant records from the available pre-query history pool and supplies the retrieved records directly as prediction context, without first summarizing them into a user profile. Thus, unlike Similarity-Ordered, Retrieve changes which historical records enter the context rather than only their order.

LLM-UM. This baseline uses a language model to summarize user information into a profile that conditions intent prediction. It compares profile-based user modeling with direct use of historical examples.

HIM-Agent. This method uses hierarchical intent memory to organize historical behavior and inform subsequent intent prediction. It provides a structured-memory comparison for modeling implicit user intent.

Query-Only Adapter. This learned baseline adapts prediction using the current query without user history. It measures improvements available from shared prediction adaptation alone.

Learned-History and Deterministic Residual. LEARNED-HISTORY learns a history-conditioned representation, while DETERMINISTIC RESIDUAL provides a deterministic residual-adaptation baseline. These comparisons examine learned history use and residual modeling alongside SAIL's probabilistic user-state formulation.

Per-User LoRA. This baseline represents user-specific low-rank parameter adaptation. It provides a parameter-based personalization comparison alongside methods that condition predictions on history or a user representation.

For history-dependent baselines without a separately defined no-history mode, we report $S = 0$ as "–" rather than equating the method with Population. Methods with explicit Population-recovery behavior retain the Population score.

Matched residual controls. MATCHED HISTORY RESIDUAL (MHR) uses the same chronological support, historical screenshot and text representations, frozen backbone, module architecture, and residual prediction head as SAIL. It also uses the same objectives, optimizer, and training schedule. We retain candidate generation but replace the Population reference in the residual and precision pathways with a stop-gradient zero vector. The residual becomes $d_i = o_i$, and the precision network receives $[h_i; o_i; \mathbf{0}; |o_i|]$.

MHR retains SAIL's 83.11M trainable parameters, but setting r_i to zero leaves its corresponding first-layer weights unused. The 3584×3584 block contains 12,845,056 weights, so approximately 70.27M parameters contribute to the forward computation. MHR therefore matches SAIL's architecture and training budget, with a smaller active parameter count.

MHR-ACTIVECAP accounts for this difference by removing the r_i input block and widening the first precision layer to 4,751 units:

$$[h_i; o_i; |o_i|] \in \mathbb{R}^{3H} \to 4751 \to 256.$$

Its precision network has 52,304,015 parameters, compared with 52,301,568 in SAIL. The total is 83,115,408 active parameters, within 0.01% of SAIL. We use the same reparameterization for the no-reference-precision condition in Appendix C.7.

Inputs and comparison scope. The input-parity audit confirms that SAIL and Recent-S at $S \in \{1, 2, 4\}$ use identical query instances and frozen support IDs. Their input representations differ: SAIL encodes historical screenshots, while the external-adaptation pipeline uses mainly textual time, scenario, and intent fields. Some external selection features also use application metadata. These comparisons measure end-to-end performance under the same chronological support budgets.

MHR uses the same observable support and module architecture as SAIL; MHR-ActiveCap also brings the active parameter count within 0.01% by widening the precision network. Equal parameter counts do not imply identical function classes, and these comparisons do not identify causal user preferences. Appendix C.7 distinguishes the questions addressed by each comparison.

Intent metrics. Following FingerTip-20K, we compute query-level intent similarity as

$$\text{Sim1} = \frac{S_1 + S_2}{2},$$

where S_1 is the cosine similarity between MiniLM embeddings of the predicted and reference intents, and S_2 is their normalized Levenshtein similarity. Our primary metric, User-Macro Sim1 ($\times 100$), first averages scores within each user and then gives every user equal weight. The component macro scores use the same user weights, so their displayed average is consistent with the reported User-Macro Sim1 up to the displayed precision. Query-Micro Sim1 instead gives equal weight to queries.

Format-normalized and slot-level diagnostics. We recompute Sim1 after applying the same deterministic format normalization to every method. This procedure changes only benchmark-prescribed surface form; it does not insert, delete, or correct semantic slot values. We also parse intents into application, action, and object/target slots and report both per-slot and joint accuracy. We fix the parsing rules before evaluating model outputs. Appendix C.2 reports the results.

GUI execution metrics. We measure task success with SR_2 and trajectory personalization with

$$\mathrm{Sim}_2 = S_{\mathrm{I}}/S_{\mathrm{II}},$$

which compares the generated action sequence with the target user's trajectory relative to a matched different-user trajectory. On successful tasks, Step Ratio divides the agent's execution steps by the user's reference steps; lower values indicate more efficient execution. We also report inference time and token usage.

Statistical analysis. For mechanism analyses, we report paired user-level 95% bootstrap confidence intervals from 10,000 resamples and Holm-adjusted p-values. These intervals measure variation across evaluation users. Mean±std entries in Table 1 instead report the mean and sample standard deviation over three independent runs. Trainable methods are reinitialized and retrained in each run with the data split, optimization schedule, and evaluation protocol fixed. For non-trainable methods, the reported dispersion is across the three independent evaluation runs, not across retraining. Methods that exactly recover the frozen Population predictor at $S = 0$ share its deterministic score and have no training-run standard deviation.

Execution aggregation. GUI execution is evaluated on the same 400 paired tasks for every method. We use one rollout per task, weight all tasks equally, and count failures or timeouts as unsuccessful executions. SR_2 is therefore the percentage of successful tasks. We report SR_2 to two decimal places so the displayed values map directly to integer success counts over the 400 tasks. Appendix C.3 specifies the task-paired estimator used for these execution comparisons. Table 22 separately varies the Population candidate-sampling seed five times at a fixed SAIL checkpoint, measuring sampling variability without retraining. The active-parameter-matched two-factor results are reported in Table 17. Persona2Web uses its benchmark-specific personalization and task-success metrics.

Zero-support recovery. At $S = 0$, all SAIL-family runs reproduce the saved Population predictions on all 1,633 evaluation queries because $\alpha_u = 0$ disables the residual pathway. This result concerns decoded outputs, not bitwise equality of logits or intermediate hidden states.

A.2 Data Provenance and Leakage Audit

Evaluation boundary. The primary within-user protocol allows a user's identity to appear in population-level training data. For each evaluation query, however, we exclude the query episode, its target, and all later interactions from training, checkpoint selection, support construction, and cached features used for that prediction. We check both episode overlap and query-relative chronology across these data sources.

Episode identifiers. We map Population-training, Population-selection, SAIL-training, and evaluation records to their original FingerTip-20K episode identifiers before checking overlap. User identities may recur across partitions under the within-user protocol, but evaluation episodes may not enter upstream training or selection. Likewise, later interactions may not influence an earlier query's prediction. All overlap statistics use the original episode identifiers, including those associated with derived records.

Split construction and model selection. We assign episode identifiers before constructing the splits. The auxiliary $60/10/13$ user split is used for Population-prior development, whereas the primary evaluation set is defined by the within-user temporal protocol. After fixing the 1,633-query evaluation set, we enforce the same episode-level and query-relative temporal boundary when forming Population training, Population selection, SAIL training, and their derived caches. Each evaluation history contains only earlier interactions from the corresponding user.

For each evaluation user, all records from that user used for Population training, Population checkpoint selection, or SAIL training precede that user's earliest evaluation query. This temporal boundary is enforced when constructing the training and selection sets, not only when filtering support at inference.

Table 5: Episode-level and temporal leakage checks. Eval overlap counts episodes shared with the 1,633 evaluation queries. Future exposure counts violations of the query-relative temporal boundary.

Data source	Size	Eval overlap	Future exposure
Population train	14,417	0	0
Population selection	2,239	0	0
SAIL train	14,284	0	0
Primary evaluation	1,633	–	–

Hyperparameter sweeps, including Figure 11, use a training-side development partition disjoint from the 1,633 evaluation queries. We fix the hyperparameters before final evaluation. The split manifest records the user IDs, episode IDs, and split-construction seed for each stage. The auxiliary Population split is not used as the sole user-boundary check for SAIL's trainable modules. The whole-pipeline user-disjoint protocol in Appendix C.4 is tied to the corresponding model checkpoints rather than inferred from the primary temporal audit.

Evaluation inputs. For all 1,633 queries from the 83 evaluation users, we check query and user identifiers, support membership, chronological ordering, and nested histories across $S \in \{1, 2, 4, 8, 20\}$. Every support interaction precedes its query; neither the query target nor later interactions enter the available history.

Training and checkpoint selection. Let E denote the 1,633-query evaluation set. At the episode level, we verify

$$E \cap D_{\text{PopTrain}} = \varnothing, \qquad E \cap D_{\text{PopSelect}} = \varnothing, \qquad E \cap D_{\text{SAILTrain}} = \varnothing.$$

SAIL uses the final checkpoint from its fixed training schedule and has no separate checkpoint-selection set. We also check that no post-query interaction contributes to the support history, Population references, SAIL training records, or cached features used for an evaluation prediction.

Profiles and screenshots. The sanitized profile, query time, and scenario contain only information available under the benchmark protocol before the target is revealed. Query screenshots show GUI states observable at prediction time; we exclude post-target and post-execution screenshots. Screenshots from the same interaction retain the same original episode identifier in these checks.

Cached references and checkpoints. Each generated intent candidate and cached Population reference remains linked to its source episode and the adapter that produced it. All downstream SAIL features in the evaluation use the frozen adapter described in the hash in Appendix B.1. The artifact manifest also records the SAIL checkpoint, split version, and prediction-file hash. These links extend the evaluation-boundary checks to preprocessing and model selection.

B Model Implementation and Optimization

B.1 Population Prior: Training Data and Optimization

Role of the Population model. The Population model supplies SAIL's shared prediction reference and its output at $S = 0$. We train it before the personalization modules and keep it frozen throughout SAIL training and evaluation. Downstream personalization therefore does not update the shared backbone.

Population-prior training split. We use an auxiliary $60/10/13$ train/validation/test user split for Population-prior development. After enforcing the primary episode-level and query-relative temporal boundary from Appendix A.2, the reported Population-training set contains 14,417 episodes and checkpoint selection uses a fixed set of 2,239 validation queries. The auxiliary test partition contains 3,144 episodes and does not participate in training or selection.

The auxiliary user split and the primary temporal evaluation serve different purposes but obey the same evaluation boundary. After mapping records to the original FingerTip-20K episode identifiers,

Table 6: Population SFT configuration.

Setting	Value
Backbone	UI-TARS-7B-SFT
SFT input modality	Text only; no history ($S = 0$)
LoRA targets	q, k, v, o projections
LoRA rank / scale / dropout	16 / 32 / 0.05
Trainable parameters	10,092,544
Epochs	3
Micro-batch / accumulation	1 / 16
Micro-steps / optimizer steps	43,251 / 2,706
Optimizer	AdamW
Learning rate / weight decay	1×10^{-4} / 0.01
Schedule	5% warmup, linear decay
Precision / gradient clipping	bfloat16 / 1.0
Random seed	42

Table 7: Population training dynamics. Dev Sim1 is used only for Population checkpoint selection and is shown on the raw scale before multiplication by 100.

Epoch	Micro-steps	Updates	Train loss	Val. loss	Dev Sim1
1	14,417	902	2.3065	3.0666	0.4514
2	28,834	1,804	1.6710	**3.0527**	0.4545
3	43,251	2,706	**1.4386**	3.1117	**0.4581**

none of the 1,633 primary evaluation queries overlaps with Population training, Population selection, or SAIL training, and no post-query interaction enters an upstream record or feature used to predict an earlier query. Appendix A.2 details these checks.

Population inputs. Each training example contains the sanitized user profile, query time, and scenario available under the benchmark protocol, excluding the target interaction and information from later behavior. The Population LoRA is optimized on text-only examples, without support history or screenshots ($S = 0$). The catalog's current-app field remains metadata and does not enter the training prompt. At downstream inference, the selected adapter is loaded onto the pretrained multimodal UI-TARS backbone. Its frozen visual encoder continues to process the available GUI screenshots under Appendix A.2's input restrictions; text-only adaptation does not make the runtime text-only. The target is the reference intent expressed as one Chinese sentence. We optimize the autoregressive language-model objective over target-intent tokens, masking prompt and padding tokens.

Parameter-efficient adaptation. We use UI-TARS-7B-SFT with the Qwen2-VL architecture. LoRA adapters modify only the text decoder's query, key, value, and output projections; the visual encoder, token embeddings, language-model head, and other backbone parameters remain frozen. Across 28 decoder layers, this gives 112 adapted linear modules and 10,092,544 trainable parameters. Table 6 lists the configuration.

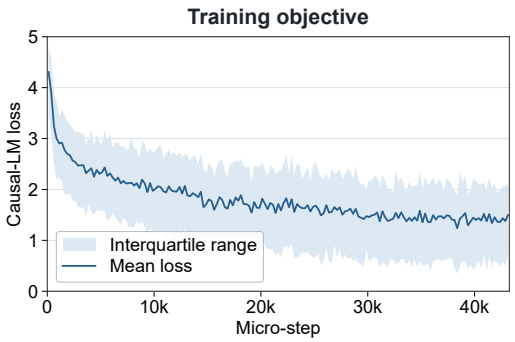


Figure 7: Population-model training dynamics.

Optimization and checkpoint selection. We train for three epochs with AdamW, a peak learning rate of 10^{-4}, weight decay of 0.01, 135 warmup updates, and linear decay. We accumulate gradients over 16 micro-batches and clip them to unit norm. After each epoch, we evaluate the checkpoint on the 2,239 validation queries with greedy decoding and select the checkpoint with the highest User-Macro Sim1.

The mean training loss falls from 2.3065 to 1.4386, and development Sim1 improves at every epoch (Figure 7; Table 7). Epoch 2 has the lowest validation cross-entropy, but epoch 3 has the highest User-Macro Sim1, our predefined selection criterion.

Training records. The log records all 43,251 micro-steps with finite losses. Mean loss decreases from 4.4010 over the first 100 steps to 1.3741 over the last 100. The selected `best` adapter is byte-identical to the epoch-3 and last adapters . We retain the configuration, training and validation logs, optimizer and scheduler states, checkpoints, and artifact manifest.

B.2 Implementation Details of SAIL

SAIL extracts frozen backbone representations and learns projection heads and query-dependent gates. The implementation below specifies the modules in Sections 3.2 and 3.4.

Representation extraction. All representations use the final hidden layer of the frozen Population checkpoint: UI-TARS-7B-SFT with the selected Population LoRA adapter loaded. Its hidden dimension is $H = 3584$. The backbone and adapter are frozen, and their extracted representations are detached. We use no multi-layer aggregation or learned layer selection.

For a support interaction (x_i, y_i), h_i is the causal hidden state that predicts the first response token. We obtain the observed-intent representation o_i by teacher-forcing the completed interaction and taking the final-layer state at its last non-padding token. Candidate representations v_{im} use the same teacher-forced encoding and pooling rule.

During training, h_q is the final-layer causal state immediately before the first supervised response token. During inference, it is the final-layer state at the last non-padding token of the initial query prompt. Both identify the state used to predict the first response token. The step-dependent $h_{q,t}^{(0)}$ is the frozen decoder state predicting token t, under the teacher-forced prefix during training and the generated prefix at inference. The Population checkpoint remains frozen for all support, candidate, observed-intent, and query encodings.

Mapping heads. The user-state, posterior-memory, and residual-interaction dimensions are

$$D_z = D_m = R = 256.$$

The support-direction network f_δ implements

$$f_\delta : \mathbb{R}^{3584} \to \mathbb{R}^{3584} \to \mathbb{R}^{256},$$

with GELU between two bias-free linear layers. The network receives the normalized direction $\bar{d}_i = d_i / \max(\|d_i\|_2, \epsilon)$; we retain residual magnitude outside the MLP:

$$\delta_i = \|d_i\|_2 f_\delta(\bar{d}_i).$$

The structured precision network follows the input definition in Section 3.2:

$$[h_i; o_i; r_i; |d_i|] \in \mathbb{R}^{4H} = \mathbb{R}^{14336}.$$

It uses the architecture

$$f_\lambda : \mathbb{R}^{14336} \to \mathbb{R}^{3584} \to \mathbb{R}^{256},$$

with GELU activation. Precision is then bounded as

$$\lambda_i = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min})\sigma(f_\lambda([h_i; o_i; r_i; |d_i|])),$$

where $\lambda_{\min} = 10^{-3}$ and $\lambda_{\max} = 100$. We initialize the output bias to give precision of approximately 0.05, avoiding near-zero support weights and early saturation.

The observability network uses

$$f_a : \mathbb{R}^{3584} \to \mathbb{R}^{3584} \to \mathbb{R}^{256},$$

with GELU between its two affine layers. Applying one element-wise sigmoid to its output gives $a_i = \sigma(f_a(h_i))$.

The posterior-memory projector receives the inferred state and its posterior variance:

$$f_m([z_u; \log \Sigma_u]) : \mathbb{R}^{512} \to \mathbb{R}^{256} \to \mathbb{R}^{256}.$$

The projector uses GELU followed by $\mathrm{LayerNorm}(256)$. At inference, $z_u = \mu_u$ and the input is $[\mu_u; \log \Sigma_u]$. The mapping heads use no dropout.

Table 8: SAIL's active trainable parameters. Counts correspond to the structured-precision and information-gated configuration.

Module	Active parameters
Residual encoder f_δ	13.76M
Structured precision f_λ	52.30M
Observability f_a	13.77M
Posterior projector f_m	0.20M
Residual head	2.10M
Dimension gate g_q	0.98M
Residual scale γ	1
Total active trainable parameters	**83.11M**

Residual-head dimensions. The four affine maps in Sec. 3.4 have input/output dimensions $H \to R$ for q_t, $D_m \to R$ for u_q, $3R \to R$ for b_t, and $R \to H$ for Δh_t. Each map includes a bias of its output dimension. For $H = 3584$ and $D_m = R = 256$, these affine maps contain 2,101,504 parameters, accounting for the 2.10M residual-head entry in Table 8. The frozen vocabulary projection is excluded from this count; the scalar γ is listed separately.

Query-conditioned gating. SAIL uses a learned dimension-wise gate and a global scale derived from posterior concentration. The dimension gate takes

$$[m_u; h_q] \in \mathbb{R}^{D_m+H} = \mathbb{R}^{3840}$$

and produces $D_m = 256$ sigmoid gates:

$$g_q = \sigma(W_g[m_u; h_q] + b_g)\,, \qquad \widetilde{m}_{u,q} = g_q \odot m_u.$$

We initialize W_g to zero and b_g to 2.0. The global scale is

$$\alpha_u = (1 - \exp(-\max(I_u, 0)))\,\mathbf{1}[S_u > 0],$$

so g_q selects the state dimensions used for a query, and α_u controls the overall correction strength. At $S = 0$, $I_u = 0$ and $\alpha_u = 0$ exactly.

Trainable parameters. With `structured` precision and `information` gating, SAIL has 83,112,961 active trainable parameters and a frozen Population backbone. Table 8 gives the module-wise breakdown of this 83.11M total.

The residual encoder uses no biases; f_λ and f_a use biased linear layers. The posterior projector includes affine $\mathrm{LayerNorm}(256)$. These modules sum to 83.11M active trainable parameters.

B.3 Training and Optimization Details

Each SAIL-family run uses a fixed budget of 17,800 microsteps with microbatch size 4, corresponding to 71,200 sampled query episodes. The training logs record 4,450 optimizer updates and no skipped updates. The Population checkpoint remains frozen. Table 9 summarizes the recorded diagnostics.

Training and evaluation. We evaluate the final checkpoint from this fixed schedule. The 1,633-query primary evaluation set is used only after training, never for optimization or selection of checkpoints or variants.

Independent training runs. Table 1 reports three independent runs. For trainable methods, each run reinitializes and retrains the trainable components while fixing the data split, Population backbone, hyperparameters, training budget, checkpoint rule, and evaluation protocol. The resulting mean and standard deviation measure run-to-run variation. Within each fixed-checkpoint mechanism or sampling diagnostic, the designated trained SAIL checkpoint is held fixed rather than averaging the three trained models. This does not imply that every diagnostic uses the same experimental run; the gate diagnostics are distinguished in Appendix C.7. Figures 8–10 show representative trajectories for different variants; they are separate from these repetitions. The five-seed candidate-sampling

experiment also addresses a different source of variation, since it fixes the trained checkpoint. Table 17 reports three-run mean point estimates. Table 18 adds the corresponding across-run standard deviations for SAIL and MHR-ActiveCap.

Optimization configuration. We train SAIL with AdamW at learning rate 10^{-4}, weight decay 0.01, 5% warmup, microbatch size 4, configured gradient accumulation 4, bfloat16 precision, and unit-norm gradient clipping. Training uses the fixed microstep budget above, with episodic support budgets sampled from $S \in \{0, 1, 2, 4, 8, 16, 20\}$. Population-reference construction uses $M = 8$ candidates, temperature 0.8, top-$p = 0.95$, and $\tau_p = \tau_c = 1.0$. Greedy decoding generates at most 64 new tokens. We set $D_z = D_m = R = 256$.

Ablation definitions. W/O POP.-REL. removes population-relative subtraction:

$$d_i = o_i, \qquad \delta_i = \|o_i\|_2 f_\delta\left(\frac{o_i}{\max(\|o_i\|_2, \epsilon)}\right). \tag{21}$$

We still generate Population candidates and pass the reference r_i to the precision network. Gaussian aggregation, observability, the global scale, dimension gate, and residual head remain unchanged.

MATCHED HISTORY RESIDUAL retains the multimodal support cache, module architecture, candidate-generation computation, and training schedule. It replaces the Population reference in the prediction pathway with $\mathbf{0}$:

$$d_i = o_i, \qquad f_\lambda([h_i; o_i; \mathbf{0}; |o_i|]),$$

Gaussian aggregation, observability, the memory projector, dimension gate, residual head, and objectives remain unchanged. The zero reference leaves 12,845,056 first-layer weights unused, giving MHR the same architecture and training budget as SAIL but fewer active parameters (Appendix A.1).

MHR-ACTIVECAP removes the zeroed reference coordinates and uses a $3H \to 4751 \to 256$ precision network. Its precision-network parameter count differs from SAIL's by 2,447, and every input block contributes to the forward computation. All other modules, support episodes, objectives, and training settings are identical to MHR.

MEAN AGGREGATION ONLY replaces the prediction-state mean with the arithmetic mean of valid support residuals, retaining Σ_u, I_u, and α_u. FIXED VARIANCE INPUT retains μ_u, I_u, and α_u but supplies the prior variance to f_m. COUNT GATE ONLY retains μ_u and Σ_u and replaces the global scale with $\alpha_u = S_u/(1+S_u)$. These controls separately test aggregation, variance input, and global gating.

COMPOSITE MEAN-STATE replaces the Gaussian prediction state with an arithmetic mean and fixed prior variance. Let $m_i^{\text{sup}} \in \{0, 1\}$ indicate a valid support interaction; padding and invalid support have mask zero. Then

$$S_u = \sum_i m_i^{\text{sup}}, \qquad \mu_u^{\text{mean}} = \frac{\sum_i m_i^{\text{sup}} \delta_i}{\max(1, S_u)}, \qquad \Sigma_u^{\text{mean}} = \lambda_0^{-1}\mathbf{1}. \tag{22}$$

It uses the count-based quantities $I_u^{\text{count}} = \log(1 + S_u)$ and $\alpha_u = S_u/(1 + S_u)$ for global scaling. Learned precision and observability remain in the support-modeling pathway and $\mathcal{L}_{\text{support}}$, but do not weight the prediction-state mean. The sums above range over valid support entries; padded entries have $m_i^{\text{sup}} = 0$ and do not contribute. The same valid-entry convention applies to the Gaussian sums in Eq. (7).

W/O DIM. GATE removes the learned affine-sigmoid dimension gate:

$$g_q = \mathbf{1}, \qquad \widetilde{m}_{u,q} = m_u. \tag{23}$$

The global scale and token-level query-dependent residual interaction remain active. This ablation removes the dimension gate's 983,296 parameters while retaining query conditioning elsewhere in the model.

The Active params column counts trainable parameters used under each variant's objective; the Population backbone remains frozen. For Composite Mean-State, Var. input is the fixed prior variance,

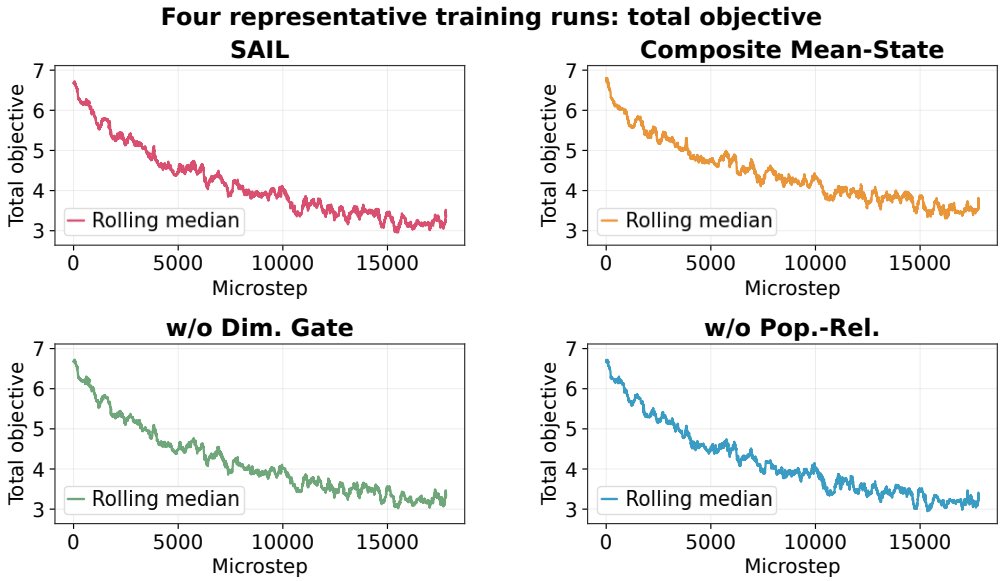


Figure 8: Total-loss trajectories across representative matched-budget runs. Run-level traces for SAIL, Composite Mean-State, w/o Dim. Gate, and w/o Pop.-Rel. under the fixed training budget.

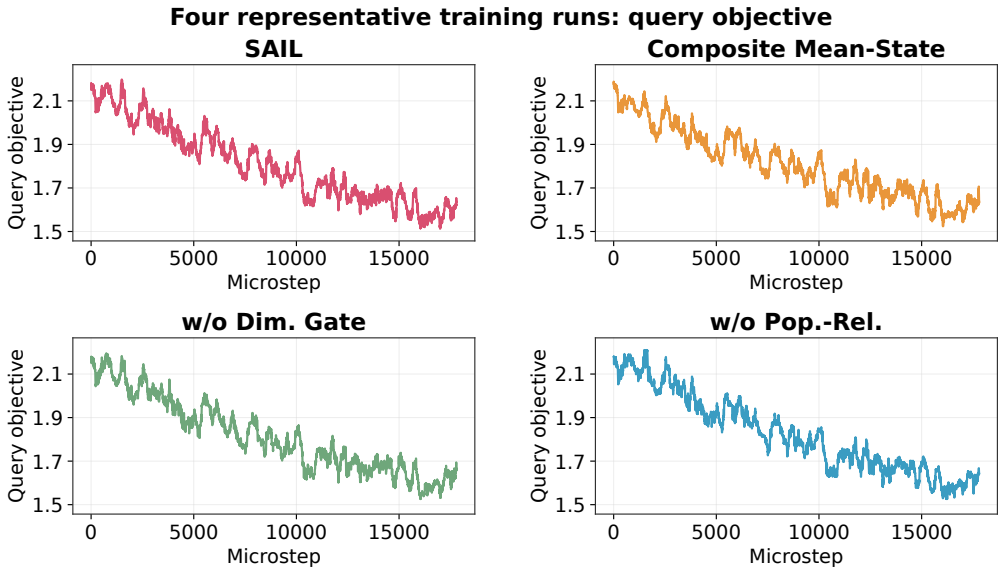


Figure 9: Query-loss trajectories across representative matched-budget runs. Query-objective traces for the same four variants shown in Figure 8.

1.000, and Gate stat. is the training mean of $\log(1+S)$, 1.561. The latter is a count statistic and does not follow the Gaussian posterior-concentration identity.

All variants use Eq. (17) with $(\lambda_q, \lambda_e, \lambda_d) = (1.0, 0.1, 1.0)$. Composite Mean-State retains learned precision and observability for $\mathcal{L}_{\text{support}}$, although neither weights its prediction mean.

Hyperparameter sensitivity. Figure 11 varies the candidate count M, user-state dimension D_z, and support-loss weight λ_e. Performance improves as M and D_z increase from restrictive settings, then remains relatively stable. The support-loss weight has a clearer interior optimum: too little support supervision weakens user-state learning, while too much can compete with query prediction. The main experiments use $M = 8$, $D_z = 256$, and $\lambda_e = 0.1$.

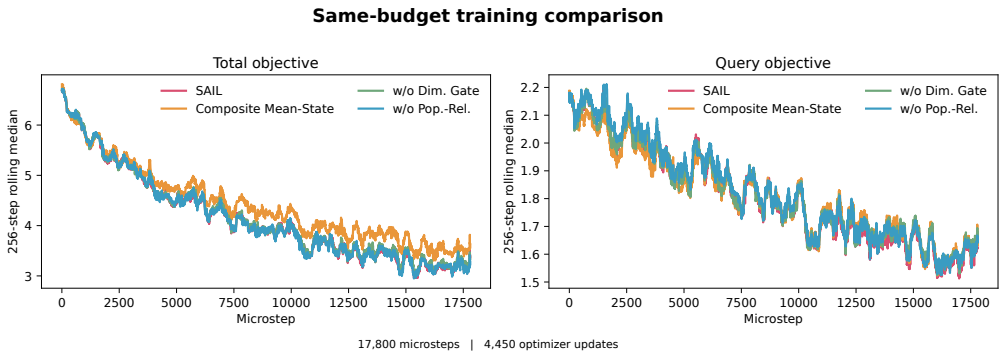


Figure 10: Same-budget training comparison. Overlay comparison for SAIL, Composite Mean-State, w/o Dim. Gate, and w/o Pop.-Rel. under 17,800 microsteps and 4,450 logged optimizer updates. The composite losses describe optimization and do not rank downstream prediction quality.

Table 9: Training diagnostics for SAIL-family runs. Var. input is the mean variance supplied to the memory projector. Gate stat. is the global scaling statistic: posterior concentration for Gaussian variants and $\log(1+S)$ for Composite Mean-State.

Variant	Microsteps	Opt. steps	Active params	Precision	Var. input	Gate stat.	Pop. grad
SAIL	17,800	4,450	83.11M	72.50	0.239	2.129	0
Composite Mean-State	17,800	4,450	83.11M	5.13	1.000	1.561	0
w/o Dim. Gate	17,800	4,450	82.13M	69.18	0.231	2.120	0
w/o Pop.-Rel.	17,800	4,450	83.11M	82.47	0.195	2.372	0

B.4 Population-Anchored Personalization with a Model-Implied User Residual

Prediction-space view. The support history defines the user-state distribution

$$q_\phi(z_u \mid \mathcal{H}_u^S) = \mathcal{N}(\mu_u, \mathrm{diag}(\Sigma_u)),$$

with parameters from Eq. (7). At inference, $m_u = f_m([\mu_u; \log \Sigma_u])$, followed by

$$\widetilde{m}_{u,q} = g_q \odot m_u, \qquad \alpha_u = (1 - \exp(-\max(I_u, 0)))\,\mathbf{1}[S_u > 0].$$

The query gate g_q selects state dimensions, and α_u scales the correction according to posterior concentration. The output logits are

$$\boldsymbol{\ell}_u(y \mid x, H_u^S) = \underbrace{\boldsymbol{\ell}_{\mathrm{pop}}(y \mid x)}_{\text{Population}} + \underbrace{\rho\, \alpha_u\, R_\theta(x, \widetilde{m}_{u,q})}_{\text{posterior-gated residual}}, \tag{24}$$

where $\rho = \mathrm{softplus}(\gamma) > 0$ is the learned residual scale. When $S = 0$, $I_u = 0$ and Eq. (24) gives the Population prediction exactly. For nonempty history, I_u measures the concentration of the modeled state distribution; it is not a calibrated probability of a correct prediction.

Interpreting the ablations. Removing population-relative subtraction lowers one-shot performance by 6.80 points (Holm-adjusted $p < 0.001$) while retaining the reference input to precision. This comparison measures subtraction's contribution within that configuration. Composite Mean-State differs from SAIL by 6.30 points at $S = 1$, but changes aggregation, variance input, and global scaling together. The separate controls in Appendix C.7 distinguish these components. Removing g_q gives a smaller one-shot difference of 1.45 points ($p_{\mathrm{Holm}} = 0.084$). The zero-history test confirms Population recovery, and additional support narrows the state distribution without increasing the online decoder context.

B.5 Incremental Updates and Interpretation of Posterior Concentration

Posterior concentration. Multiplying the learned factors in Eq. (6) with the prior gives $q_\phi(z_u \mid \mathcal{H}_u^S)$. Its variance describes the spread of the user state under this factor model. With $\Sigma_{0,j} = \lambda_0^{-1}$,

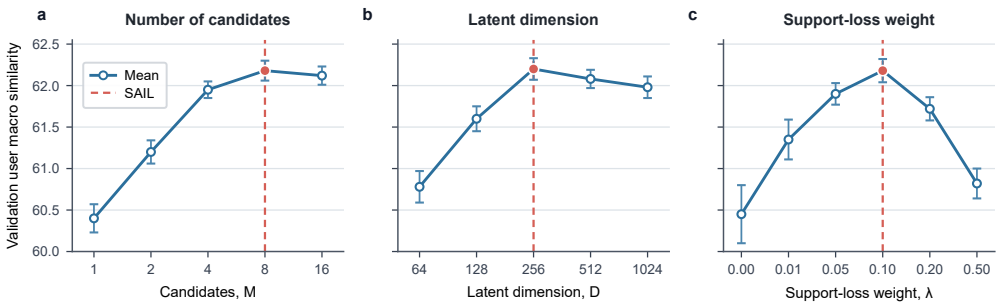


Figure 11: Hyperparameter sensitivity of SAIL. Left: Population candidate count M. Middle: user-state dimension D_z. Right: support-loss weight λ_e. Performance stabilizes as M and D_z grow, while λ_e has a clearer interior optimum. Dashed lines mark the main-experiment settings. Error bars show standard deviation over three independent runs for each setting, separate from the main Table 1 repetitions.

we can write

$$I_u = \frac{1}{2D_z} \sum_{j=1}^{D_z} \log \frac{\Sigma_{0,j}}{\Sigma_{u,j}}.$$

Larger I_u means a narrower distribution. It does not by itself imply a more accurate user model or better predictions. Figure 5 visualizes departure from the prior and contraction in this model-implied sense.

Incremental aggregation. Let $\eta_u = \sum_i \lambda_i \odot a_i \odot \delta_i$. Then $\mu_u = \eta_u \oslash \Lambda_u$ and $\Sigma_u = \mathbf{1} \oslash \Lambda_u$, where $\oslash$ denotes element-wise division. With the network and cached factors fixed, a new interaction updates

$$\Lambda'_u = \Lambda_u + \lambda_{\text{new}} \odot a_{\text{new}}^{\odot 2}, \qquad \eta'_u = \eta_u + \lambda_{\text{new}} \odot a_{\text{new}} \odot \delta_{\text{new}}.$$

These sums give the updated mean and variance without re-encoding earlier interactions. Reordering a fixed collection of factors leaves the sums unchanged. The histories themselves must still satisfy the chronological support constraints.

Monotonic variance reduction. Because $\lambda_{\text{new},j} > 0$ and $a_{\text{new},j}^2 \geq 0$,

$$\Lambda'_{u,j} = \Lambda_{u,j} + \lambda_{\text{new},j} a_{\text{new},j}^2 \geq \Lambda_{u,j}, \qquad \Sigma'_{u,j} \leq \Sigma_{u,j}, \qquad I'_u \geq I_u. \tag{25}$$

Adding a fixed factor therefore cannot decrease posterior concentration. Since α_u is nondecreasing in I_u, adding such a factor also cannot reduce the global adjustment scale, even for repeated or conflicting observations. These are properties of the aggregation rule, not guarantees of evidence quality.

Repeated, conflicting, and zero-residual history. Each factor adds $\lambda_i \odot a_i^2$ to precision, regardless of its residual's sign. Repeating an interaction adds precision without a new observation. Opposing residuals can also cancel in the mean while increasing precision. For example, in one dimension, let $\lambda_0 = \lambda_1 = \lambda_2 = 1$, $a_1 = a_2 = 1/2$, and $\delta_1 = 1$, $\delta_2 = -1$. Then

$$\Lambda_u = \frac{3}{2}, \qquad \mu_u = 0, \qquad \Sigma_u = \frac{2}{3}, \qquad I_u = \frac{1}{2} \log \frac{3}{2} > 0.$$

The mean is zero, yet the variance is smaller than the prior variance of one. This is an algebraic example, not an experimental result. Setting every δ_i to zero likewise keeps $\mu_u = 0$ while allowing precision to grow. In each case, increasing concentration need not add independent or useful information about the user.

Effect of the support objective. For fixed fit error $e = \delta_{i,j} - a_{i,j} z_{u,j}$, the precision-dependent loss is $\frac{1}{2}(\lambda e^2 - \log \lambda)$ up to a positive averaging constant. Holding e fixed gives

$$\frac{\partial}{\partial \lambda} \frac{1}{2} (\lambda e^2 - \log \lambda) = \frac{1}{2} \left(e^2 - \frac{1}{\lambda} \right).$$

Table 10: Detailed metric breakdown for SAIL through $S = 8$. All similarity metrics are on a 0–100 scale. Query-level Sim1 is rounded before aggregation, while S_1 and S_2 are aggregated from unrounded component scores; the displayed macro values need not satisfy an exact averaging identity.

S	Query-Micro Sim1	User-Macro Sim1	S_1 Macro	S_2 Macro	Exact Match	Invalid Format
0	50.37	50.61	62.33	38.89	0.43	6.80
1	63.09	62.35	70.70	54.00	7.59	0.00
2	62.87	62.05	70.42	53.68	7.78	0.00
4	63.15	62.43	70.73	54.12	8.08	0.00
8	63.34	62.57	70.94	54.20	8.27	0.00

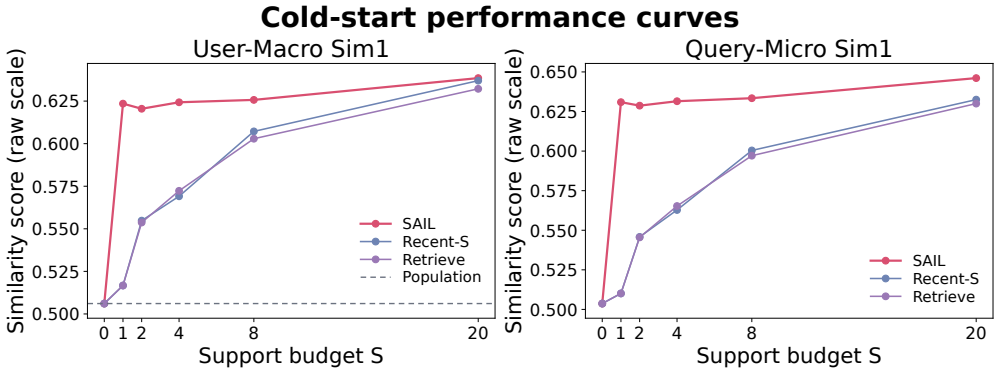


Figure 12: Cold-start performance across support budgets. User-Macro Sim1 and Query-Micro Sim1 trajectories for SAIL and representative baselines across support sizes. Scores are on the raw scale before multiplication by 100.

A sufficiently small fit error favors higher precision. In training, precision is bounded and the model learns both error and precision, so this partial derivative describes only one part of the optimization. It explains how fitting the learned residuals can increase concentration without establishing that the user model is correct. Appendix C.7 tests the support objective and compares instance-dependent global scaling with a fixed support-budget mean.

Exact Population recovery. At $S = 0$, the posterior equals the prior and $I_u = \alpha_u = 0$, so the residual in Eq. (16) vanishes. With nonempty history, a zero mean is insufficient: the memory projector also receives $\log \Sigma_u$, and the residual pathway contains learned affine terms. RMS normalization also removes raw residual magnitude from the final scale away from its numerical floor. Neither concentration nor raw residual size alone therefore forces a zero correction. Appendix C.9 measures prediction changes and negative transfer directly.

C Extended Quantitative Results

C.1 Full Metric Breakdown

Table 10 reports SAIL's intent metrics through $S = 8$, including the primary User-Macro Sim1. Most of the gain appears with the first few interactions. SAIL remains strong across the low-shot regime and reaches 63.85 User-Macro Sim1 at $S = 20$; the intermediate scores are not monotonic.

SAIL's advantage over Recent-S is largest with limited history: the margins at $S = 1, 2, 4, 8, 20$ are $+10.68$, $+6.57$, $+5.52$, $+1.85$, and $+0.15$ points. Relative to the improvement from $S = 0$ to $S = 20$, the budgets $S = 1, 2, 4, 8$ recover 88.7%, 86.4%, 89.3%, and 90.3% of that gain. Thus, a few interactions capture most of the benefit, and $S = 20$ achieves the highest displayed score without a monotonic increase at every budget.

C.2 Format-Normalized and Slot-Level Evaluation

SAIL produces no invalid-format outputs once support is available in Table 10. To separate formatting improvements from semantic accuracy, we normalize each method's outputs with the same

Table 11: Format-normalized and slot-level evaluation at $S = 1$. The same deterministic canonicalizer is applied to every method.

Method	Norm. Sim1	App Acc.	Action Acc.	Object Acc.	Joint
Population	50.90	71.6	58.4	43.2	28.5
Deterministic Residual	60.35	80.6	72.8	56.9	38.7
Matched History Residual	61.18	81.4	74.1	58.3	40.1
SAIL	**62.44**	**82.7**	**76.0**	**60.5**	**42.0**

Table 12: Task-matched replacement. Correct uses the target user's earlier interactions; Other uses task-matched history from another user. Both satisfy the chronological constraints. Matching does not use the held-out target.

S	Correct	Other	Δ
1	62.18	58.27	+3.91
2	61.91	57.86	+4.05
4	62.29	57.43	+4.86

deterministic procedure, recompute Sim1, and score application, action, and object/target slots separately. Normalization changes delimiters, whitespace, and benchmark-prescribed surface form, but leaves slot values unchanged.

SAIL retains its advantage after format normalization and across all three slot types. Its gains therefore extend beyond the removal of malformed outputs.

C.3 Task-Matched Replacement and Extended GUI Execution Results

Task-matched replacement. We replace the target user's support with task-matched history from another user, keeping the same budget S and chronological constraints. Matching uses only information available at prediction time, excluding the held-out intent y_q and later interactions. This changes the user associated with the history while retaining task relevance as closely as possible. We evaluate queries with a valid match; the Correct scores therefore differ slightly from SAIL's full-set scores in Table 1.

Correct-user support outperforms task-matched other-user support by 3.91, 4.05, and 4.86 User-Macro Sim1 points at $S = 1, 2, 4$ (Table 12). The user's own history retains predictive information that a task-matched substitute loses. This tests the value of user identity under task matching, without establishing a causal separation of task and user factors.

Extended GUI execution comparison. Table 13 extends the main-paper comparison to a broader set of history-aware baselines at $S = 4$. Every method uses the same 400 paired execution tasks and fixed executor. The executor receives each method's predicted intent, neither the reference intent nor SAIL's latent user state. The displayed SR_2 differences are 2.50 percentage points over Retrieve and 1.25 over MHR.

Execution estimator and paired resampling. Each of the 400 tasks is executed once per method and receives equal weight. Let $b_{m,i} \in \{0, 1\}$ denote success for method m on task i; failures and timeouts are assigned zero. We compute

$$\widehat{\mathrm{SR}}_2(m) = \frac{100}{400} \sum_{i=1}^{400} b_{m,i}.$$

For a comparison with baseline m, define the task-paired difference $d_{m,i} = b_{\mathrm{SAIL},i} - b_{m,i}$ and $\widehat{\Delta}_m = 100 \sum_i d_{m,i}/400$. A paired percentile-bootstrap calculation resamples 400 task IDs with replacement, retaining the two methods' outcomes together. Repeating this 10,000 times gives bootstrap differences $\widehat{\Delta}_m^{*(1)}, \ldots, \widehat{\Delta}_m^{*(10000)}$; the interval endpoints are their empirical 2.5th and 97.5th percentiles. This resampling preserves the within-task dependence between the compared methods.

Table 13: Extended GUI execution results on FingerTip-20K. History-aware methods use $S = 4$ on the same 400 paired execution tasks.

Method	$SR_2 \uparrow$	$Sim_2 \uparrow$	Step↓	Time↓	Tok.↓
Population	38.50	1.05	1.23	10.18	2456
Recent-S	39.50	1.10	1.23	10.31	2615
Similarity-Ordered	40.00	1.11	1.22	10.35	2601
Me-Agent HPM	40.00	1.12	1.24	10.48	2674
MemoryBank	40.25	1.13	1.22	10.42	2655
HIM-Agent	40.50	1.14	1.21	10.50	2648
LLM-UM	39.50	1.11	1.25	10.58	2690
Retrieve	40.75	1.15	1.21	10.47	2632
MHR	42.00	1.20	1.20	10.50	2528
SAIL	**43.25**	**1.25**	**1.18**	10.53	2510

C.4 Full User-Disjoint and Cross-Benchmark Results

Tables 14 and 15 give the complete comparisons for user-disjoint FingerTip-20K evaluation and Persona2Web transfer.

Whole-pipeline user boundary. The user-disjoint split contains 60/10/13 train/dev/test users and 260 held-out test queries. Every globally learned SAIL module and every learned control is trained using only the 60 training users, while hyperparameter selection uses only the training/development users. The Population checkpoint follows the same held-out-user boundary.

At evaluation, SAIL infers each held-out user's state only from chronologically valid support interactions. For Per-User LoRA, any test-time update uses that support alone and never the held-out target. For reproducibility, we retain the training-user set, development-user set, checkpoint identity, cache provenance, and test prediction file for each reported model. Thus the reported user-disjoint results reflect a whole-pipeline held-out-user evaluation rather than only a user-disjoint Population prior.

Transfer protocol and uncertainty. For FingerTip-20K, we bootstrap held-out users so that queries from the same user remain in one cluster. For Persona2Web, we transfer the FingerTip-trained checkpoint and its hyperparameters without retraining or model selection. Each of 50 personas supplies four support interactions and one held-out target. We compute confidence intervals by persona-clustered bootstrap resampling.

Metric computation and aggregation. The Persona2Web columns summarize different outcomes: P_{web} measures website personalization, P_{pref} measures preference matching, and Intent measures task accuracy apart from personalization. These scores are reported on a 0–100 scale; Success is reported as a percentage. Table 15 reports the full results, with paired transfer differences summarized below.

Cross-domain input construction. The four completed support interactions form the historical input, while the held-out target is reserved for evaluation. In SAIL's notation, the support must distinguish each historical context x_i from its observed behavior y_i rather than use the held-out target as a support label.

Paired transfer differences. On user-disjoint FingerTip-20K, SAIL exceeds MHR by $+1.20$ points at $S = 1$ (95% CI $[+0.48, +1.92]$), $+1.05$ at $S = 2$ ($[+0.34, +1.76]$), and $+0.95$ at $S = 4$ ($[+0.22, +1.68]$). These are the reported held-out-user differences under the whole-pipeline boundary described above. The MHR-ActiveCap row gives $+1.13/+0.99/+0.88$ points at $S = 1, 2, 4$. On Persona2Web, the differences are $+1.5$ for P_{web} ($[+0.3, +2.7]$), $+1.6$ for P_{pref} ($[+0.2, +3.0]$), $+0.4$ for Intent ($[-0.1, +0.9]$), and $+2.0$ percentage points for Success ($[-1.8, +5.8]$). The smaller persona-level sample gives wider intervals. The personalization metrics provide clearer evidence of user-model transfer than the task-success difference, whose interval crosses zero.

Table 14: Full same-benchmark user-disjoint evaluation. The split contains 60/10/13 train/dev/test users and 260 held-out test queries. Dashes follow the same no-history reporting convention as Table 1.

Method	$S = 0$	$S = 1$	$S = 2$	$S = 4$
Population	48.72	48.72	48.72	48.72
Recent-S	–	49.16	51.38	53.97
Similarity-Ordered	–	49.16	51.52	54.21
Query-Only Adapter	**51.63**	51.63	51.63	51.63
Learned-History	–	53.84	55.29	56.43
Per-User LoRA	–	54.55	55.41	56.20
Deterministic Residual	48.72	54.92	56.08	57.01
Matched History Residual	48.72	55.27	56.33	57.21
MHR-ActiveCap	48.72	55.34	56.39	57.28
SAIL	48.72	**56.47**	**57.38**	**58.16**

Table 15: Full cross-benchmark transfer on Persona2Web. Fifty personas are evaluated with $S = 4$ support interactions. P_{web} measures whether the agent identifies a website consistent with recurring user-history patterns; P_{pref} measures whether it identifies items or attributes matching user preferences. Intent measures task accuracy apart from personalization, and Success requires both correct personalization and task execution.

Method	P_{web}	P_{pref}	Intent	Success
No History	20.0	11.5	48.0	0.0
Query-Only Adapter	20.8	12.5	50.0	0.0
Recent-S	34.4	25.0	48.4	2.0
Similarity-Ordered	36.8	28.0	49.2	2.0
Learned-History	41.2	33.5	50.4	4.0
Per-User LoRA	42.8	35.1	50.7	4.0
Deterministic Residual	44.0	36.5	50.7	6.0
Matched History Residual	44.9	37.4	50.8	6.0
SAIL	**46.4**	**39.0**	**51.2**	**8.0**

C.5 Broader Baseline Behavior

History-based baselines improve as more interactions become available, narrowing SAIL's advantage. At $S = 20$, SAIL reaches 63.85 User-Macro Sim1, slightly above the strongest direct-history baseline at 63.70. Its largest relative gains occur with sparse history. Performance remains strong across the evaluated budgets, with the highest score at $S = 20$.

C.6 User- and Query-Level Heterogeneity

SAIL's limited-history gains extend to most users, with varying magnitudes. At $S = 1$, the mean and median gains over Recent-S are $+10.69$ and $+10.35$ Sim1 points, and 90.4% of users improve. The user-level win rates are 79.5% at $S = 2$ and 73.5% at $S = 4$. Figure 2 shows the paired per-user distributions; the average gain is not driven by only a few users.

Query-level heterogeneity. Figure 4 shows paired query-level differences $\text{Sim1}_{\text{SAIL}} - \text{Sim1}_{\text{baseline}}$ at $S \in \{1, 2, 4\}$ against Population, Recent-S, and Similarity-Ordered. Positive values favor SAIL and negative values favor the reference method.

The distributions favor positive gains, most strongly at $S = 1$, but retain negative tails. Increasing support from $S = 1$ to $S = 4$ reduces the positive shift as direct-history methods become more competitive. This variation across queries also motivates averaging within users before reporting User-Macro Sim1, so users with more queries do not dominate the result.

C.7 Mechanism Diagnostics Under Limited History

The ablations test individual operations and combinations of changes. Unless stated otherwise, all variants use the same frozen backbone, support protocol, and training budget. Subtraction and dimension-gate ablations each remove one operation while leaving the other pathways intact. Composite Mean-State changes aggregation, variance input, and global scaling together. Appendix B.3

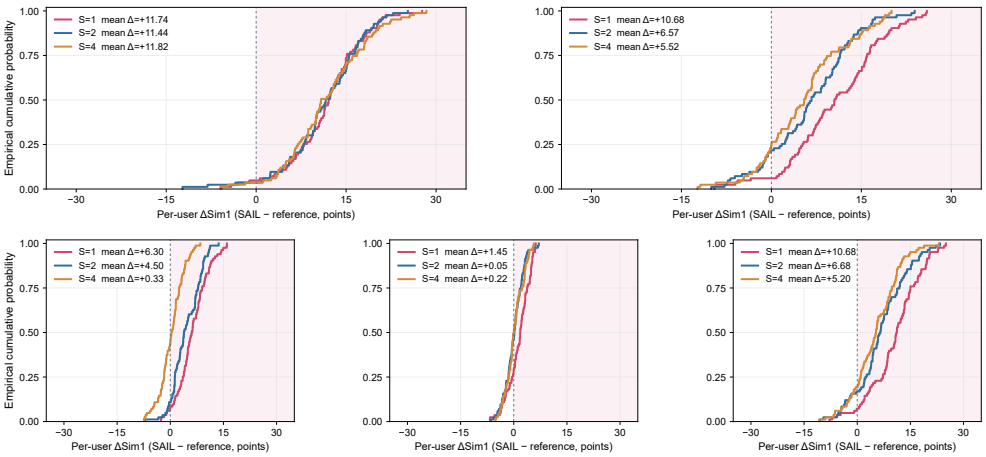


Figure 13: Per-user gain distributions. Empirical cumulative distributions of paired user-level Sim1 differences at $S \in \{1, 2, 4\}$ against Population, Recent-S, the Composite Mean-State control, the no-dimension-gate variant, and Retrieve. Positive values favor SAIL; the full distributions retain both positive and negative tails.

Table 16: Mechanism ablations under limited history. Δ@1 denotes full SAIL minus the corresponding ablation at $S = 1$. Confidence intervals and Holm-adjusted p-values use paired user-level bootstrap resampling.

Method	$S = 1$	$S = 2$	$S = 4$	Δ@1	**95% CI**	p_{Holm}
SAIL w/o Pop.-Rel.	55.55	58.10	61.85	+6.80	[+5.21,+8.39]	< 0.001
Composite Mean-State	56.05	57.55	62.10	+6.30	[+4.76,+7.85]	< 0.001
Mean Aggregation Only	60.88	61.19	62.12	+1.47	[+0.82,+2.11]	0.002
Fixed Variance Input	61.72	61.63	62.26	+0.63	[+0.12,+1.14]	0.031
Count Gate Only	61.11	61.52	62.10	+1.24	[+0.64,+1.84]	0.004
SAIL w/o Dim. Gate	60.90	62.00	62.21	+1.45	[-0.10,+3.00]	0.084
SAIL	**62.35**	**62.05**	**62.43**	–	–	–

defines the controls that vary these three components separately. Table 16 reports scores and one-shot paired user-bootstrap statistics. These intervals measure variation across evaluation users; Table 1 separately reports variation across three training runs.

Population-relative subtraction. Removing subtraction lowers Sim1 by 6.80 points at $S = 1$ (95% CI $[+5.21, +8.39]$, Holm-adjusted $p < 0.001$), followed by 3.95 and 0.58 points at $S = 2, 4$. This variant sets $d_i = o_i$ but still supplies Population candidates and r_i to the precision network. The differences therefore measure subtraction's effect with that reference pathway retained. The bootstrap interval describes evaluation-user variation; training-run variation is reported separately where three-seed repetitions are available.

Subtraction and reference-conditioned precision. Table 17 varies residual subtraction and the explicit reference input to precision independently. SAIL-NoRefPrec-ActiveCap retains $d_i = o_i - r_i$ and supplies $[h_i; o_i; |d_i|]$ to a $3H \to 4751 \to 256$ precision network. The reference still affects the residual and its magnitude, but has no separate precision-input block. MHR-ActiveCap instead uses $d_i = o_i$ and removes both reference pathways. The wider precision network matches active parameter counts without making the function classes identical.

Table 17: Two-factor control for residual subtraction and reference-conditioned precision. Cells without an explicit reference block use the active-parameter-matched $3H \to 4751 \to 256$ precision network. Scores are three-run mean point estimates. The SAIL and MHR-ActiveCap rows reuse the same means as Table 18, which also reports their across-run standard deviations.

Variant	Subtract r_i	r_i in f_λ	Active params	$S=1$	$S=2$	$S=4$
SAIL	✓	✓	83.11M	**62.35**	**62.05**	**62.43**
SAIL-NoRefPrec-ActiveCap	✓	–	83.12M	61.68	61.74	62.18
w/o Pop.-Rel.	–	✓	83.11M	55.55	58.10	61.85
MHR-ActiveCap	–	–	83.12M	60.08	60.61	61.30

At $S = 1$, the subtraction difference is $62.35 - 55.55 = 6.80$ with an explicit precision reference and $61.68 - 60.08 = 1.60$ without it. Conversely, adding the explicit precision reference changes the score by $+0.67$ with subtraction and -4.53 without it. These contrasts suggest configuration-dependent interaction rather than independent additive effects. The 6.80-point ablation therefore measures subtraction within the reference-conditioned precision configuration, not its standalone contribution.

Table 18: Active-capacity-matched training-seed results. Mean±std is reported over three independently seeded runs. The final row is the difference of the displayed three-run means; no paired run-level significance test is claimed.

Method	$S=1$	$S=2$	$S=4$
SAIL	62.35±0.42	62.05±0.36	62.43±0.33
MHR-ActiveCap	60.08±0.40	60.61±0.38	61.30±0.34
Difference of displayed means	+2.27	+1.44	+1.13

Independent-run protocol. SAIL and MHR-ActiveCap are evaluated with three paired training seeds while fixing the split, Population checkpoint, candidate cache, optimization budget, final-checkpoint rule, and evaluation queries. We retain the seed identifiers, checkpoints, and user-level predictions; repeated evaluation passes from one trained checkpoint are not counted as independent training runs.

For method m, run k, and support budget S, we first average each user's query scores and then average equally over users to obtain $a_{m,k,S}$. We report the mean and sample standard deviation of $a_{m,k,S}$ across the three independently trained runs. Conditional evaluation uncertainty is estimated separately by resampling users jointly for both methods while retaining their queries and seed-specific predictions; repeated predictions for the same user are not treated as independent users. With $K = 3$, the reported summaries are

$$\bar{a}_{m,S} = \frac{1}{K}\sum_{k=1}^{K} a_{m,k,S}, \qquad s_{m,S} = \sqrt{\frac{1}{K-1}\sum_{k=1}^{K}(a_{m,k,S} - \bar{a}_{m,S})^2}.$$

The final row of Table 18 is the difference of the displayed method means. Across-run variability is reported separately for the two methods, and no run-level significance test is claimed.

What each comparison measures. At $S = 1, 2, 4$, Table 1 gives gains of $+10.68/+6.57/+5.20$ over the strongest prior history-based methods ($+7.48$ on average). These are end-to-end differences at common support budgets. The gains over MHR are $+2.43/+1.54/+1.25$ ($+1.74$ on average), and those over Deterministic Residual are $+2.14/+1.22/+0.51$ ($+1.29$ on average).

The MHR-ActiveCap comparison in Table 17 gives $+2.27/+1.44/+1.13$ ($+1.61$ on average). This control compares population-relative construction under matched observable inputs, active parameter counts, and training budgets. The independent-run results follow the protocol above and are reported in Table 18. We therefore report the $+7.48$ end-to-end gain and the $+1.61$ average matched-control gain as complementary quantities: the former measures the complete system against prior methods, whereas the latter asks what advantage remains under tightly matched adaptation conditions. Neither is a decomposition of the other or a pure causal effect of an individual component.

Scale-matched observation control. To test whether subtraction mainly changes representation scale, this control retains the no-explicit-reference precision architecture $3H \to 4751 \to 256$ and compares three factor inputs:

$$d_i^{\text{rel}} = o_i - r_i, \qquad d_i^{\text{raw}} = o_i, \qquad d_i^{\text{scale}} = c\, o_i, \qquad c = \sqrt{\frac{\sum_{i \in \mathcal{D}_{\text{train}}} \|o_i - r_i\|_2^2}{\max(\sum_{i \in \mathcal{D}_{\text{train}}} \|o_i\|_2^2, 10^{-12})}}.$$

The scalar c is fixed from the frozen training cache before any control is trained; it is not fitted using evaluation users or targets. All three conditions feed $[h_i; o_i; |d_i|]$ to precision, retain the same residual encoder and objectives, and are trained with matched budgets and paired seeds. This matches global input RMS, not every higher-order feature statistic or function class. The reference is used to set the training scalar only; the scaled-observation condition does not access r_i in its per-instance prediction pathway.

Table 19: Scale-matching observation control. All values are User-Macro Sim1 ($\times 100$).

Factor input; no explicit reference in precision	$S = 1$	$S = 2$	$S = 4$
Population-relative $o_i - r_i$	61.68	61.74	62.18
Raw observation o_i	60.08	60.61	61.30
Scale-matched observation $c\, o_i$	60.96	61.27	61.91

The scale-matched observation row provides the corresponding comparison under matched global input RMS. This control does not match every higher-order feature statistic or function class and does not identify causal user preferences.

Observation-conditioned reference weighting. We compare four candidate-weighting rules: uniform averaging, Population likelihood alone, observation compatibility alone, and their joint use in Section 3.2. All variants use the same candidates, residual pathway, and training budget.

Table 20: Observation-conditioned Population-reference weighting. All values are User-Macro Sim1 ($\times 100$).

Reference construction	$S = 1$	$S = 2$	$S = 4$
Uniform candidate mean	60.74	61.05	61.92
Likelihood only	61.08	61.28	62.03
Compatibility only	61.71	61.66	62.21
Full likelihood + compatibility	**62.35**	**62.05**	**62.43**

The comparison favors joint likelihood and compatibility weighting, with the largest differences at one shot. It examines how SAIL constructs the reference, separately from the effect of having a reference at all.

Composite Mean-State control. SAIL exceeds Composite Mean-State by 6.30 points at $S = 1$ (95% CI $[+4.76, +7.85]$, Holm-adjusted $p < 0.001$), 4.50 at $S = 2$, and 0.33 at $S = 4$. This control jointly replaces precision-weighted aggregation with an arithmetic mean, fixes the variance input, and sets $\alpha_u = S/(1 + S)$. The 6.30-point difference therefore reflects all three changes and cannot be assigned to posterior weighting alone.

Precision controls. As supplementary comparisons, fixed precision ($c = 1$) reaches 57.42/59.76/61.67 User-Macro Sim1 at $S = 1, 2, 4$, while global precision ($c = 32$) reaches 60.03/60.86/62.06. We do not use these controls as the primary evidence for individual mechanism effects.

Support objective and global gate. The support loss in Eq. (19) can favor higher precision even for small residuals. We therefore test both the auxiliary objective and the use of instance-dependent correction strength. The $\lambda_e = 0$ variant retrains SAIL without $\mathcal{L}_{\text{support}}$. The fixed-$\alpha$ variant retains the trained state and residual pathway but replaces each query's α_u with its training-split mean at the same support budget.

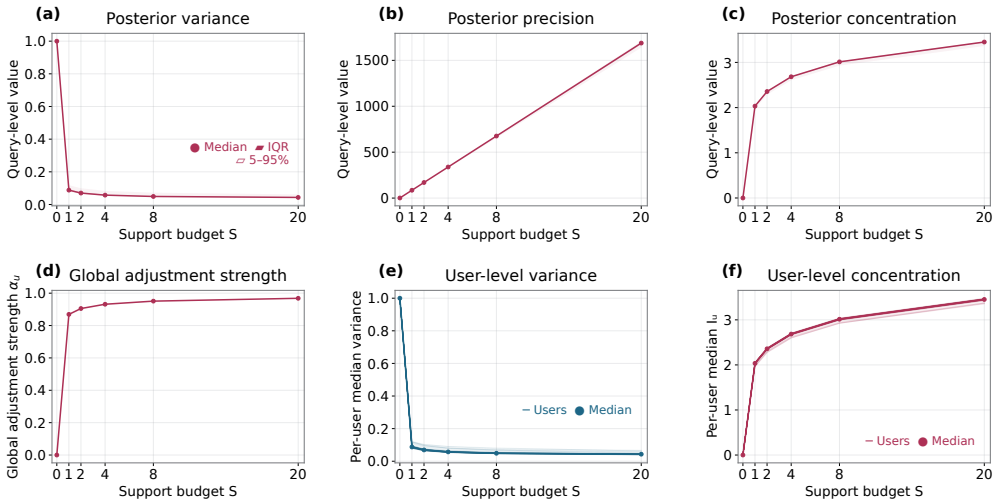


Figure 14: User-state statistics at the SAIL checkpoint used for this diagnostic. (a)–(d) show query-level medians of posterior variance, precision, concentration I_u, and global adjustment strength α_u; shading marks the IQR and 5–95% range. (e)–(f) show trajectories for 83 users and their median. I_u describes the model state, not calibrated correctness. These are summaries within this diagnostic, not alternative aggregations of the records in Table 21.

Table 21: Support-objective and global-gate diagnostic at $S = 4$. NTR is measured relative to Population on the natural test queries.

Setting	Sim1 ↑	mean λ	mean I_u	mean α_u	NTR ↓
$\lambda_e = 0$	61.66	31.8	1.48	0.69	19.2%
Fixed mean α	61.95	88.6	2.36	0.84	18.1%
SAIL	**62.43**	88.6	2.36	0.84	**14.6%**

Removing $\mathcal{L}_{\text{support}}$ tests how auxiliary factor training affects precision and concentration. Fixing α_u keeps the inferred state unchanged and tests whether history-dependent correction strength reduces negative transfer compared with the support-budget mean. Both comparisons measure prediction behavior directly, without treating I_u as calibrated reliability.

Gate summaries and figure provenance. The scalar α_u in Eq. (12) and the vector g_q are different quantities. When a scalar summary of the dimension gate is used, we denote it explicitly as $\bar{g}_q = D_m^{-1} \sum_j g_{q,j}$. For each query, $\alpha_u = 1 - \exp(-I_u)$ is computed after Gaussian support aggregation and before taking cross-query means or quantiles; in general $\mathbb{E}[\alpha_u] \neq 1 - \exp(-\mathbb{E}[I_u])$. The logged field `gate_alpha` denotes the scalar global strength α_u, not the dimension gate g_q.

The gate statistics in Figure 6, Table 21, and Figures 14–15 are not all computed from the same experimental run. Each diagnostic reports statistics within its own experimental setting, rather than alternative summaries of a shared evaluation sample. Figure 14 reports medians and quantiles, whereas Table 21 reports means. The table evaluates support-objective and global-scale variants on natural queries at $S = 4$; the figures separately examine posterior-state statistics across support budgets and candidate-disagreement groups. Accordingly, their absolute values are interpreted within the corresponding diagnostic rather than as repeated estimates from one shared evaluation sample.

Dimension-wise query gating. Removing g_q lowers one-shot performance by 1.45 points (95% CI $[-0.10, +3.00]$, Holm-adjusted $p = 0.084$). The differences at $S = 2$ and $S = 4$ are 0.05 and 0.22 points. These results suggest a role in selecting dimensions for the current query, with limited evidence that the gate drives the average gain.

Population-candidate disagreement. Candidate sampling can produce different references for the same GUI context. We examine how disagreement among candidates relates to reference construction, query-conditioned personalization, and prediction quality. For interaction i, we measure

Table 22: Sensitivity to Population-candidate sampling. User-Macro Sim1 is the mean and standard deviation over five candidate-sampling seeds at the same trained SAIL checkpoint. Reference and posterior similarity are mean pairwise cosine similarities across these realizations. Range is the maximum minus minimum Sim1. This experiment does not vary the training seed.

S	User-Macro Sim1	Ref. Sim.	Post. Sim.	Range
1	62.31 ± 0.11	0.958	0.972	0.29
2	62.02 ± 0.09	0.961	0.979	0.24
4	62.39 ± 0.07	0.964	0.984	0.19

disagreement by the mean pairwise cosine distance between candidate representations:

$$D_i = \frac{2}{M(M-1)} \sum_{m<m'} \left(1 - \cos(v_{im}, v_{im'})\right). \tag{26}$$

Larger D_i means less agreement among the sampled intents. We use empirical tertiles for the low-, medium-, and high-disagreement summaries below. A separate diagnostic in Figure 15 groups candidate disagreement into quintiles.

We sort the normalized candidate weights in descending order,

$$\rho_{i(1)} \geq \rho_{i(2)} \geq \cdots \geq \rho_{i(M)}, \tag{27}$$

and summarize the reference mass with

$$R_i^{(1)} = \rho_{i(1)}, \qquad R_i^{(3)} = \sum_{j=1}^{3} \rho_{i(j)}, \qquad R_i^{(\text{rest})} = 1 - R_i^{(3)}. \tag{28}$$

Reference weighting under disagreement. As candidate disagreement increases from low to medium to high, mean Top-1 mass rises from 0.173 to 0.179 and 0.189. Top-3 mass similarly rises from 0.470 to 0.483 and 0.508. Candidate-weight entropy falls from 2.035 to 2.028 and 1.996, and the effective candidate count falls from 7.68 to 7.61 and 7.37. SAIL thus places modestly more weight on fewer compatible candidates while continuing to combine several alternatives.

Downstream response to disagreement. Mean learned precision changes little across the three groups: 89.06, 88.65, and 88.37 from low to high disagreement. Together with the candidate weights, these values show more selective reference construction and a small reduction in precision as disagreement grows. They describe model behavior, but do not calibrate when personalization should be weakened. The separate experiment in Figure 15 shows quintile-level patterns in candidate allocation, global adjustment strength, and prediction quality.

Stability across candidate-sampling seeds. We regenerate Population candidates with five independent sampling seeds while fixing the trained SAIL checkpoint designated for this sampling experiment, support histories, evaluation queries, and decoding settings. Each realization reconstructs the references, residual factors, posterior states, and final predictions. This tests how much candidate sampling alone changes SAIL's limited-history performance.

For each fixed support interaction, reference stability is the mean pairwise cosine similarity across sampling realizations:

$$\text{Stab}_r = \frac{2}{K(K-1)} \sum_{a<b} \cos\left(r_i^{(a)}, r_i^{(b)}\right), \tag{29}$$

where $K = 5$. We define posterior-state stability in the same way:

$$\text{Stab}_\mu = \frac{2}{K(K-1)} \sum_{a<b} \cos\left(\mu_u^{(a)}, \mu_u^{(b)}\right). \tag{30}$$

Table 22 reports representation and prediction stability.

Reference cosine similarity ranges from 0.958 to 0.964 across sampling realizations, and posterior-state similarity ranges from 0.972 to 0.984. The aggregated states are therefore more stable than the

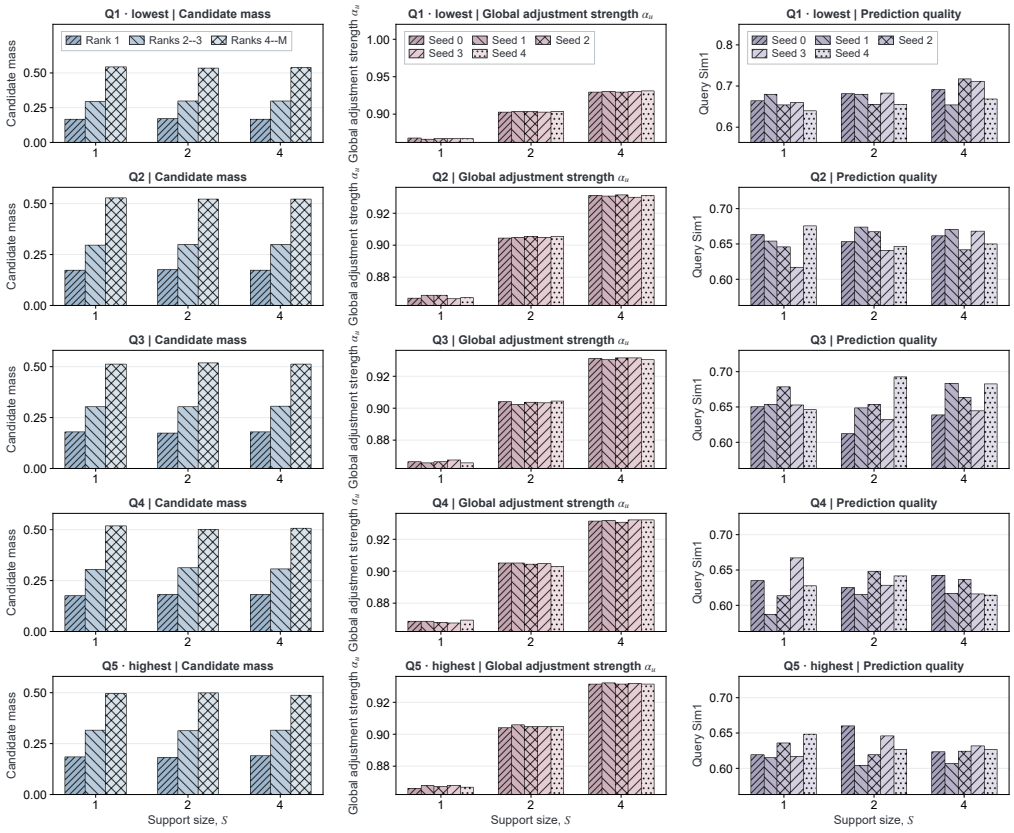


Figure 15: Model behavior by Population-candidate disagreement. Rows show increasing disagreement quintiles from Q1 (lowest) to Q5 (highest). Columns show candidate-reference allocation, global adjustment strength α_u, and prediction quality at $S \in \{1, 2, 4\}$ for the checkpoint held fixed within this diagnostic. This is a separate experiment from the tertile summaries above. Hatches distinguish candidate-rank groups and independent candidate-sampling realizations, respectively.

sampled references. At this checkpoint, Sim1 standard deviations are 0.11, 0.09, and 0.07 points at $S = 1, 2, 4$, with ranges of 0.29, 0.24, and 0.19 points.

This experiment measures candidate-sampling variation for a fixed SAIL model. It does not estimate variation in SAIL-minus-control differences across independent training runs or provide training-seed confidence intervals. The disagreement analysis likewise describes reference weighting and precision; it addresses a different source of variation from the training-run results in Table 1.

Mechanism summary. The subtraction ablation measures a 6.80-point one-shot difference with the precision-reference pathway retained. The active-parameter-matched two-factor comparison shows that the observed subtraction benefit depends on the precision configuration. The additional scale-matching control tests one alternative explanation. The reference-weighting comparison addresses likelihood and observation compatibility within reference construction.

Gaussian aggregation uses learned precision and observability to combine historical deviations. The support-loss and fixed-α comparisons show that training shapes concentration and that instance-dependent scaling can reduce negative transfer relative to a support-budget average. The dimension gate further selects how the state applies to the current query.

Greater candidate disagreement is associated with a modest concentration of weight on compatible alternatives. Across sampling realizations, posterior states vary less than references, and prediction scores change little. These findings describe the roles of the learned quantities; they do not make precision, concentration, candidate weights, or correction strength calibrated probabilities of prediction correctness or user specificity.

C.8 Controlled Behavioral Attribution and Gain–Harm Evaluation

Diagnostic design. The primary benchmark measures predictive performance but leaves the source of behavioral differences unknown. We therefore construct an evaluation-only diagnostic that varies the user rule and environment independently. Its ground truth follows from the construction, rather than another model's judgment.

The diagnostic contains 320 held-out queries, with 80 per condition, from 20 predefined behavioral-rule profiles and eight GUI templates. All conditions use $S = 4$ support interactions. We use none of these instances to train Population, SAIL, MHR, Query-Only, or selection scores. The rule inventory contains four target-free choices: select the highest-rated, lowest-priced, or most recent option, or reuse a saved/familiar source. Rule identities are fixed before evaluation.

The four conditions vary the source of the observed behavior. (i) *User varies / environment fixed*: the rule changes while GUI rendering, semantic options, and option order remain fixed. (ii) *User fixed / environment shifts*: layout, option order, and application skin change while the rule and semantic choice set remain fixed. (iii) *Environment-forced*: the interface permits only one valid action, leaving no user-discriminative choice. (iv) *Shared Population error*: all profiles use the same target rule, and we retain contexts where Population selects the same wrong alternative in at least three of four renderings. Ground truth selects this held-out subset only; it is unavailable to the evaluated methods at inference.

These conditions distinguish a difference from Population from behavior that transfers across environments. They also test whether corrections depend on user history or mainly repair shared prediction errors.

Controls. Query-Only Adapter tests shared prediction correction without user history. MHR uses SAIL's multimodal support, module architecture, residual head, supervision, and training budget, but removes the population-relative reference. Its zeroed reference block leaves fewer active parameters; Appendix C.7 includes MHR-ActiveCap to examine this difference. These controls compare shared correction, history-conditioned adaptation, and population-relative modeling. We measure prediction shift with the mean token-level Jensen–Shannon divergence between personalized and Population next-token distributions under the same teacher-forced prefix. Mean α_u reports global correction strength.

Table 23: Controlled behavioral attribution. All conditions use $S = 4$ support interactions. Δ denotes User-Macro Sim1 points relative to Population. Mean α_u is SAIL's global correction strength; Shift is its mean token-level Jensen–Shannon divergence from Population under the same teacher-forced prefix.

Condition	Query-Only Δ	MHR Δ	SAIL Δ	mean α_u	Shift
User varies / environment fixed	+0.8	+5.3	**+7.6**	0.82	0.074
User fixed / environment shifts	+0.9	+2.6	**+5.5**	0.76	0.061
Environment-forced	+0.4	-0.9	+0.0	0.29	0.017
Shared Population error	**+6.1**	+4.8	+0.6	0.34	0.021

Controlled results. SAIL gains most when behavior follows a transferable user rule, including when the environment changes. With environment-forced actions, mean $\alpha_u = 0.29$ and prediction shift $= 0.017$ indicate small corrections. Under shared Population error, Query-Only captures most of the generic improvement; SAIL gains only +0.6 points, with mean $\alpha_u = 0.34$ and shift $= 0.021$. The correction measurements help distinguish reduced personalization from simply failing to match Query-Only's gain. They do not establish causal identification of user preference. These condition-level averages do not rule out negative transfer on individual queries.

Gain–harm metrics and selection scores. Average similarity can hide queries harmed by personalization. Let $S^{\mathrm{Pop}}_{u,q}$ and $S^{\mathrm{Pers}}_{u,q}$ be query-level Sim1 in $[0, 1]$ for Population and the personalized predictor. A target-free score $s_{u,q}$ and threshold t define $A_t = \mathbf{1}[s_{u,q} \geq t]$ and coverage $C(t) = \Pr(A_t = 1)$. Unselected queries use Population:

$$\widetilde{S}_{u,q}(t) = A_t S^{\mathrm{Pers}}_{u,q} + (1 - A_t) S^{\mathrm{Pop}}_{u,q}. \tag{31}$$

We compute gain and positive harm over all queries:

$$G(t) = \mathbb{E}_{\text{all}}\left[A_t\left(S^{\text{Pers}}_{u,q} - S^{\text{Pop}}_{u,q}\right)\right], \tag{32}$$

$$H(t) = \mathbb{E}_{\text{all}}\left[A_t\left(S^{\text{Pop}}_{u,q} - S^{\text{Pers}}_{u,q}\right)_+\right], \tag{33}$$

Unselected queries contribute zero to both. Among selected queries, the negative-transfer rate is

$$\text{NTR}_\epsilon(t) = \Pr\left[S^{\text{Pers}}_{u,q} < S^{\text{Pop}}_{u,q} - \epsilon \mid A_t = 1\right], \qquad \epsilon = 0.05. \tag{34}$$

Here ϵ corresponds to a five-point decrease on the 0–100 Sim1 scale in the result tables.

SAIL's selection score uses only inference-time quantities:

$$s^{\text{SAIL}}_{u,q} = \alpha_u \frac{\|g_q \odot m_u\|_2}{\|m_u\|_2 + 10^{-6}}. \tag{35}$$

The MHR score applies the same formula to its own state and gate. Task similarity is $\max_i \cos(h_i, h_q)$ over valid support interactions, and Population confidence is the exponentiated length-normalized mean token log-probability of the Population output. In the system-level comparison, task similarity and Population confidence select between Population and Recent-S predictions, the MHR score selects between Population and MHR predictions, and the SAIL score selects between Population and SAIL predictions. For 50% coverage, each score ranks the evaluation queries and selects the top half, without using target labels for ranking or thresholding. This compares gain and harm at a fixed intervention rate; it does not calibrate a deployment threshold.

Table 24: System-level gain–harm trade-off at 50% matched personalization coverage. Gain and positive harm are all-query expectations multiplied by 100; NTR is the conditional rate among selected queries using $\epsilon = 0.05$. This table compares the implemented score–prediction configurations and is not a fixed-prediction selector-only ablation.

Selection score	Gain ↑	Harm ↓	NTR ↓
Task similarity	3.10	1.55	18.8%
Population confidence	3.40	1.34	16.2%
MHR-based score	4.00	1.05	12.8%
SAIL model-implied score	**4.80**	**0.74**	**9.8%**

Gain–harm results. At the same coverage, the SAIL-based configuration gives greater gain, lower positive harm, and lower negative-transfer rate than the other implemented score–prediction configurations in Table 24. We use this result only as a system-level selective-personalization diagnostic: matched coverage controls intervention frequency, but it does not isolate the quality of the ranking score from the quality of the underlying personalized predictions. We therefore do not attribute the entire difference to the selection score itself.

Table 25: Fixed-prediction selector comparison at 50% coverage. The comparison fixes SAIL predictions and varies only the ranking score. Gain and harm are all-query expectations multiplied by 100; NTR is the percentage among selected queries at $\epsilon = 0.05$.

Selection score	Gain ↑	Harm ↓	NTR ↓
Task similarity	3.46	1.21	15.9%
Population confidence	3.73	1.10	14.1%
MHR-style score	4.18	0.93	12.0%
SAIL score	4.80	0.74	9.8%

Fixed-prediction selector protocol. The experiment uses a fixed trained SAIL checkpoint and saves its personalized prediction and the Population prediction for every query at $S = 4$. Every selector then chooses between those same two saved outputs; an MHR-derived score changes only the ranking, never the selected personalized prediction. We compare four selectors at fixed 50% coverage: task similarity, Population confidence, the MHR-derived score, and the SAIL score. Score orientations are fixed on development data rather than selected after examining test gains.

Each selector chooses the top $\lfloor N/2 \rfloor$ queries, breaking ties by a fixed target-independent query-ID order. Thus realized coverage is $\lfloor N/2 \rfloor / N$. We report all-query gain and positive harm, and selected-query NTR with $\epsilon = 0.05$, using uniform query weights. These expectations are not interchangeable with User-Macro Sim1 when users have different numbers of queries. Paired score comparisons resample users jointly, retaining all their queries. The MHR-based and MHR-style labels refer to an MHR-derived ranking score in the two different protocols: the latter chooses from fixed SAIL/Population outputs and does not use MHR predictions.

Two consistency checks prevent predictor changes from contaminating this comparison. At 100% coverage every selector recovers exactly the same SAIL outputs and metrics. At 0% coverage all queries recover Population, gain and harm are zero, and selected-query NTR is undefined. At 50% coverage, the SAIL-score entry matches the corresponding system-level configuration when the checkpoint, prediction cache, query set, ranking, and weighting are identical. These checks ensure that the comparison isolates ranking behavior from changes in the underlying personalized predictor.

Deviation–relevance landscape. Using the same trained SAIL checkpoint, we summarize each support interaction with the scalar contribution

$$w_i = \frac{1}{D_z} \sum_{j=1}^{D_z} \lambda_{i,j} a_{i,j}^2, \qquad \widetilde{w}_i = \frac{w_i}{\sum_k w_k},$$

and define

$$D_{\mathrm{pop}}(q, S) = \sum_{i=1}^{S} \widetilde{w}_i \|o_i - r_i\|_2, \qquad R_{\mathrm{query}} = 0.7\, r_{\mathrm{app}} + 0.3\, r_{\mathrm{scenario}}.$$

Here r_{app} and r_{scenario} are the support–query application-match and scenario-match rates. Neither axis uses the held-out query target. We divide both axes into equal-frequency quintiles before computing the query density, mean SAIL-minus-Recent-S gain, and negative-transfer probability.

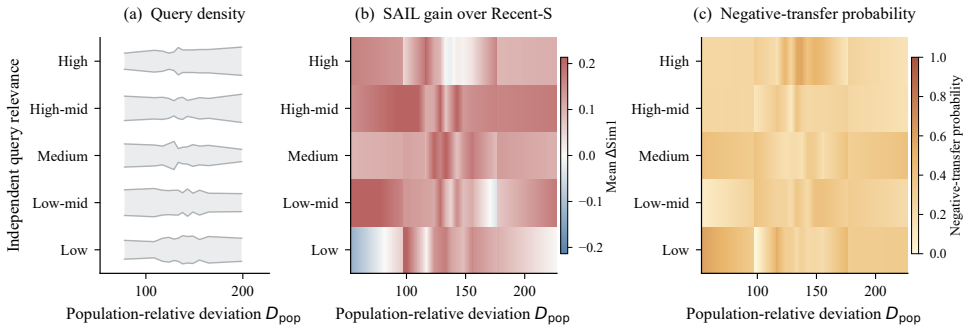


Figure 16: Personalization value landscape under sparse history. **(A)** Query density across population-relative deviation and independent relevance regimes. **(B)** Mean query-level Sim1 gain of SAIL over Recent-S. **(C)** Negative-transfer probability over the same regions. The non-monotone pattern shows that neither deviation magnitude nor task relevance alone determines personalization value.

Larger population-relative deviations do not consistently yield larger gains, and task-relevant histories can still cause negative transfer. Together with the controlled conditions, this shows why behavioral deviation, cross-context consistency, and user-specific predictive value should be examined separately.

Interpretation. The controlled conditions, matched-coverage comparison, and natural-query landscape show complementary aspects of history use. Sparse history is most useful when it predicts the target user's behavior beyond the GUI context and shared Population error. Forced actions, generic prediction errors, and superficially relevant interactions may add little user information or harm prediction. Evaluating both average gain and negative transfer captures these different outcomes.

C.9 Posterior-Concentration Stress Tests

Construction. We test whether posterior concentration corresponds to useful history under four adverse support conditions and an informative correct-user positive control, without using held-out query targets. *Population-typical* support consists of chronologically valid interactions with small residual magnitude, using thresholds fixed on the training split. *Repeated* support duplicates one valid interaction to fill the same nominal budget. *Conflicting* support pairs valid interactions with strongly negative cosine similarity between normalized residual directions. *Wrong-user* support follows the task-matched replacement protocol in Appendix C.3. Informative correct-user support provides a positive control.

Metrics and fixed-strength comparison. For each support condition, we report $\Delta\mathrm{Sim1} = \mathrm{Sim1}_{\mathrm{method}} - \mathrm{Sim1}_{\mathrm{Population}}$ and negative-transfer rate with $\epsilon = 0.05$. NTR measures the frequency of drops exceeding five Sim1 points, whereas positive harm measures loss magnitude. This table focuses on the former; it does not treat NTR as a substitute for the latter. These metrics use the same per-query scores and query weighting within each stress subset. We also report mean α_u and prediction shift, measured as mean token-level Jensen–Shannon divergence under the same teacher-forced prefix (Appendix C.8).

The fixed-strength comparator retains SAIL's inferred state, dimension gate, and residual head. It replaces each query's α_u with the training-split mean $\bar{\alpha}_S$ at that support budget, using a common training-derived scale across stress conditions rather than instance-dependent global strength.

Table 26: Posterior-concentration stress tests at $S = 4$. Δ is Sim1 points relative to Population; NTR is reported as a percentage using $\epsilon = 0.05$. Shift is token-level Jensen–Shannon divergence.

Condition	Fixed-α Δ	SAIL Δ	Fixed NTR	SAIL NTR	Fixed Shift	SAIL Shift	mean α_u
Informative correct	+8.40	+10.90	14.2	9.7	0.081	0.079	0.84
Population-typical	-1.36	-0.22	43.5	28.1	0.079	0.018	0.31
Repeated	-0.74	-0.28	39.7	31.0	0.080	0.031	0.72
Conflicting	-1.92	-0.61	48.4	34.6	0.078	0.025	0.58
Wrong-user	-2.76	-1.18	53.1	39.4	0.082	0.034	0.49

Instance-dependent gating lowers negative-transfer rate in all four adverse conditions relative to fixed strength and substantially reduces prediction shift for Population-typical, conflicting, and wrong-user histories. Repeated and conflicting history still produce nonzero corrections. SAIL thus reduces over-personalization in these tests without eliminating it, and posterior concentration remains distinct from calibrated support reliability.

D Efficiency and Execution Diagnostics

D.1 Online Context Accounting

Online input grows with history for Recent-S and Similarity-Ordered, from roughly 2.37k tokens at one shot to 3.07k at $S = 20$. SAIL uses approximately 2.34k tokens at every evaluated support budget (Table 28).

Table 27: Accuracy–compute trade-off for Population candidates. Validation results at $S = 1$ using SAIL. Support preprocessing is a one-time state-update cost; query latency measures online generation after the state has been cached.

M	User-Macro Sim1	Support prep. (ms)	Query latency (ms)
1	60.72	118	410
2	61.46	206	411
4	62.02	382	412
8	**62.35**	718	413
16	62.41	1395	414

Performance levels off near $M = 8$, while support preprocessing continues to grow almost linearly. Once the user state is cached, candidate count has little effect on query-generation latency.

Table 28: Online-input token scaling. Growth is measured from $S = 1$ to $S = 20$.

Method	S = 1	S = 2	S = 4	S = 8	S = 20	Growth
Recent-S	2371.15	2408.02	2481.67	2629.06	3072.53	+29.6%
Similarity-Ordered	2371.15	2408.02	2481.68	2629.06	3072.53	+29.6%
SAIL	2335.25	2335.25	2335.25	2335.25	2335.25	0.0%

At $S = 20$, SAIL uses 737 fewer online-input tokens than Recent-S, a reduction of approximately 24.0%. This measures the generation context only: constructing Population references and updating the state still require support-side computation. The full raw history, however, need not appear in each subsequent prompt.

Accuracy–context trade-off. SAIL exceeds Recent-S by 10.68, 6.57, 5.52, 1.85, and 0.15 User-Macro Sim1 points at $S = 1, 2, 4, 8, 20$. Its relative advantage is largest with sparse history and remains positive throughout this range. As support increases from $S = 1$ to $S = 20$, SAIL's score rises from 62.35 to 63.85 while its online input stays at approximately 2.34k tokens. Direct-history methods close the gap by adding raw interactions to a growing decoder context; SAIL retains a bounded representation and a small advantage at $S = 20$.

D.2 Offline Action-Level Personalization Diagnostics

We use device-free offline probes to examine whether same-user history contains predictive action-level information. Figure 18 uses a Similarity-S4 retrieval proxy on non-test data. For each query, support contexts are ranked by cosine similarity in the frozen context-representation space and the top four same-user interactions are used. Wrong-user controls use task-matched other-user histories selected by the same similarity rule. Action accuracy is computed on the reported action families, and the lift is $\mathrm{Acc}_{\text{same-user}} - \mathbb{E}[\mathrm{Acc}_{\text{wrong-user}}]$. Trajectory stages are defined by thirds of normalized step index; intent-similarity bins are $[0, 0.1)$, $[0.1, 0.2)$, $[0.2, 0.4)$, and $[0.4, 1]$ as shown in the figure. Error bars use 10,000 episode-clustered bootstrap resamples. This diagnostic does not measure complete action correctness or device-level task success and is separate from the end-to-end $\mathrm{SR}_2/\mathrm{Sim}_2$ evaluation.

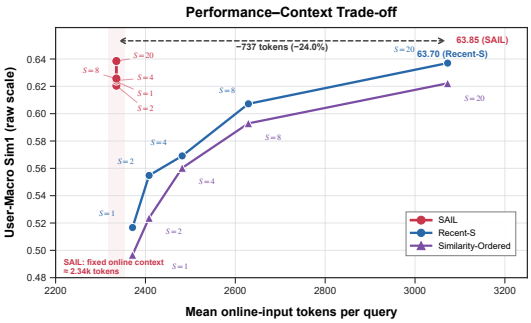


Figure 17: Performance–context trade-off. Accuracy is plotted against deterministic online input size across support budgets, highlighting the trade-off between retaining raw history and keeping a bounded online representation.

Figure 18 groups same-user accuracy gains by action family, trajectory stage, and intent similarity. Gains are largest for scrolling (+15.3 pp) and typing (+12.5 pp), with smaller positive effects for clicking, waiting, and termination. Gains appear at early, middle, and late stages and peak in intermediate-to-high intent-similarity groups.

A separate learned probe in Figure 19 provides a second test of action-level user information; it is not the Similarity-S4 retrieval proxy above. The probe concatenates a frozen query-context embedding with the mean embedding of the four support interactions, projects the result to 256 dimensions with GELU, and predicts the action family with a linear head. Same User uses the target user's support, Wrong User substitutes task-matched other-user support, and Zero History sets the support vector to zero. Generic is a separately trained query-only classifier on the same training/validation split. Same User $S = 4$ improves accuracy by $+0.70$ percentage points over Generic, $+1.38$ over Wrong User, and $+3.94$ over Zero History. The differences are modest and consistently favor the correct user's history.

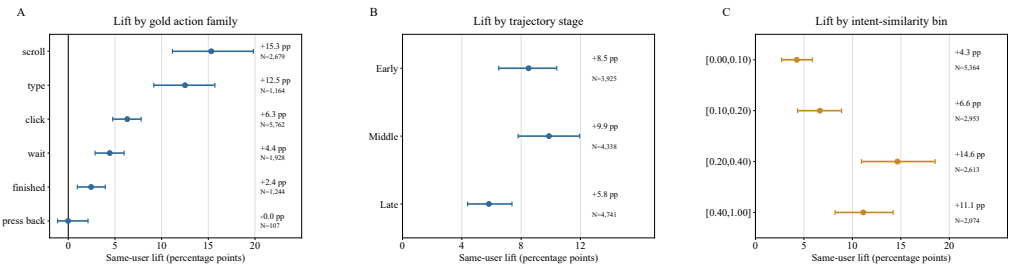


Figure 18: Offline same-user action gains for the Similarity-S4 retrieval proxy. Lift is same-user accuracy minus mean accuracy over valid task-matched wrong-user histories. Left: action family; middle: trajectory stage; right: intent-similarity bin. The left panel excludes 120 actions not mapped to the six reported families, retaining 12,884 steps; the other panels retain all 13,004. Error bars are episode-clustered bootstrap intervals. This non-test, device-free diagnostic is not SAIL checkpoint performance or real-device task success.



Figure 19: Learned offline action-level personalization probe. Left: action-family prediction under Generic, Same User, Wrong User, and Zero History conditions. Right: paired accuracy gains of Same User over each control. These probes complement the end-to-end $\mathrm{SR}_2/\mathrm{Sim}_2$ evaluation.

E Qualitative Cases and Supplementary Discussion

Purpose and overview. All Sim1 values in this subsection are on the raw scale before multiplication by 100. We inspect representative validation cases to understand when personal history improves intent prediction and when it does not. The examples cover positive transfer, negligible change, negative transfer, and a task-matched wrong-user comparison. Figure 20 summarizes these patterns, contrasting transferable routines and user-conditioned tendencies with redundant, misleading, environment-driven, and wrong-user histories. Together, they illustrate why task relevance or behavioral deviation alone is insufficient to determine whether personal history should modify the Population prediction. Figures 21–23 provide detailed examples.

Behavioral transfer and task-relevant history. Figure 21(a–b) shows two cross-application patterns. In the first, support interactions come from different applications but repeatedly select the first item; Population instead predicts a specific basketball-news action and scores 0.152 Sim1, while SAIL recovers the intended operation and scores 1.000. The second case transfers a recurring check-in routine across applications. Figure 21(c) provides a complementary task-relevant example: all four support interactions involve Xiaohongshu searches, and SAIL recovers the displayed target exactly, improving from Population's 0.555 Sim1 to 1.000.

When relevant or matched history is insufficient. Finance-related support appears relevant to the query in Figure 22, yet redirects SAIL toward an incorrect Alipay action, reducing Sim1 from 0.634 to 0.276. This shows that topical similarity does not necessarily imply user-specific predictive value. Figure 23 further compares task-matched wrong-user history with the target user's own history: the former yields 0.211 Sim1, whereas the latter recovers the target intent and scores 1.000. Thus, task matching alone does not preserve the same behavioral information as the correct user's history. These cases show that task relevance alone does not determine the value of personal history: transferable user-conditioned information can help, while redundant or superficially relevant history can mislead prediction.



Figure 20: Overview of representative qualitative personalization patterns. The examples contrast transferable user-conditioned behaviors with task-matched wrong-user configurations, redundant evidence, misleading history, and environment-driven behavior. They illustrate when sparse personal history can provide useful information beyond the Population prediction and when it should be used cautiously.

Cross-App Action Habits Can Transfer

Observed User History

1. Open Alibaba; select the first item.
2. Open Sports Live; select the first item.
3. Open Damai; search Shenzhen; select the first item.
4. Open Hema; select the first item.

Target	Open Hupu and select the first item.	
Population	NBA tab: view CBA semifinal news.	0.152
SAIL	Open Hupu and select the first item.	**1.000**

(a) Cross-application action transfer. Repeatedly selecting the first item across applications transfers to the held-out Hupu query.

Cross-App Routines Can Transfer

Observed User History

1. Open Tomato Audio; complete the daily check-in.
2. Open Amap; navigate by car to Pengcun School.
3. Open Tomato Novel; go to Benefits and check in.
4. Open Tencent Video; complete the check-in.

Target	Open Anipop and complete the daily check-in.	
Population	Open Plants vs. Zombies 2 and enter the garden.	0.175
SAIL	Open Anipop and complete the daily check-in.	**1.000**

(b) Cross-application routine transfer. Check-ins across applications provide a transferable routine for the held-out Anipop query.

Task-Relevant History Can Recover the Target

Observed User History

1. Search Xiaohongshu for UR's new spring outfits.
2. Search Xiaohongshu for travel destinations.
3. Search Xiaohongshu for a Xuzhou travel guide.
4. Search Xiaohongshu for a Universal Beijing travel guide.

Target	Search Xiaohongshu for mosquito-bite relief.	
Population	Open Xiaohongshu, search for mosquito-bite relief, and view the first recommended post.	0.555
SAIL	Search Xiaohongshu for mosquito-bite relief.	**1.000**

(c) Task-relevant history recovers the target. The support interactions all involve searches in the same application, and personalization recovers the held-out search intent exactly.

Figure 21: Positive validation cases. Panels (a)–(c) show cross-application action transfer, a recurring routine, and a task-relevant history case that recovers the held-out intent.

Surface Relevance Can Mislead

Observed User History

1. Open Douyin; view Xu Qiongna's finance news.
2. Open Zhitianxia; view the news.
3. Open Haima Babi; check the baby.
4. Open Eastmoney; view stock-market quotes.

Target	Use Tencent Yuanbao to search for U.S. stock-market opening hours.	
Population	Open Yuanbao and view U.S. market trading hours.	0.634
SAIL	Open Alipay and view stock-market quotes.	**0.276**

Figure 22: Negative transfer from superficially relevant history. Finance-related history redirects the prediction toward the wrong application and action despite its apparent relevance to the query.

Task-Matched History Is Not Evidence of the Correct User

Task-Matched Wrong-User History

1. Bilibili: play a video about *A Simple Spark*.
2. Bilibili: watch an uploader's video.
3. Bilibili: play Jay Chou's *How Are You*.
4. Bilibili: play *Can't Say It*.

Correct-User History

1. Xiaohongshu: like and save the latest recipe post.
2. Weather: view today's forecast.
3. Meituan: add the first recommendation to cart.
4. Bilibili Messages: view the latest likes.

Target	Open Bilibili; select a recommended video; like and save it.	
Population	Like the post.	0.199
SAIL + wrong-user	Open Bilibili and like an OldSao video.	0.211
SAIL + correct-user	Open Bilibili; select a recommended video; like and save it.	**1.000**

Figure 23: Task matching and correct-user history. Task-matched history from another user gives a weaker prediction, while the correct user's history recovers the target intent.