RELEVANCE DOES NOT IMPLY APPLICABILITY: EXPERIENCE ACTIVATION FOR PERSONAL GUI AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Personal Graphical User Interface (GUI) agents rely on interaction history to infer what a user wants from ambiguous instructions and to anticipate recurring routines. Existing approaches retrieve task-relevant history and append it to the policy's context, implicitly assuming that experience relevant to a task remains useful for each decision within it. We find that this help is largely spent at the first decision: retrieved history strongly improves the opening step of an episode, yet provides little sustained benefit over the remaining 90% of steps, and offers weak guidance on whether a proactive suggestion is warranted. A relevant record may tell the agent where to begin, but not which past action applies to the current screen or whether a routine is due now. The underlying issue is that *relevance does not imply applicability*: relevance is determined at the task level, whereas applicability depends on the situation at decision time. We therefore recast personalization as *experience activation* and introduce **ExpActivator**, a training-free framework that activates only the experience applicable to the current situation. During execution, ExpActivator matches each new screen to historical states in the frozen GUI backbone's latent space and supplies the corresponding action as a reference. Before execution, it activates a recurring intent only when the current time and scenario provide sufficient support, and otherwise abstains. Across four GUI backbones, ExpActivator improves within-trajectory step success by 28% on average, achieves the best personalized execution on every backbone while using about one-fifth as many history tokens, and reaches approximately 2.3× the Matthews correlation coefficient of the strongest proactive baseline. Experience pays where it is activated, not where it is appended.

1 INTRODUCTION

Personal Graphical User Interface (GUI) agents operate apps on a user's own device, where much of what the user wants is never stated explicitly (Jang et al., 2026; Yuan et al., 2026; Yang et al., 2026; Li et al., 2026). "Order my usual coffee" names neither the app, the drink, nor the store; a user who orders coffee at home every morning may even expect the agent to offer it without being asked. Such gaps can only be filled from the user's interaction history (Yang et al., 2025; Chen et al., 2026; Kim et al., 2026): in *personalized execution*, history helps resolve an ambiguous instruction and carry it out the way this user would; in *proactive suggestion*, it helps decide whether a recurring routine is due before any instruction is given.[1] The question is therefore not whether history helps, but when and how past experience should influence each decision.

Existing personalized GUI agents primarily use history through task-level retrieval: they retrieve related records, profiles, or intent memories and append them to the policy's context (Chai et al., 2026; Yang et al., 2025; Kong et al., 2026; Lyu et al., 2026). This implicitly treats task-level relevance as a sufficient proxy for decision-level applicability: once experience is retrieved for a task, the same context is expected to remain useful as subsequent decisions change. For execution, this leaves a sharper question: not only how much retrieved history helps, but for how long.

[1] Throughout the paper, a *task* is a user goal such as ordering coffee, an *episode* is one execution of a task on the device, a *step* is a single screen within an episode together with the action taken on it, and a *routine* is a task that the user repeats at similar times and in similar scenarios, such as at home or while commuting.

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

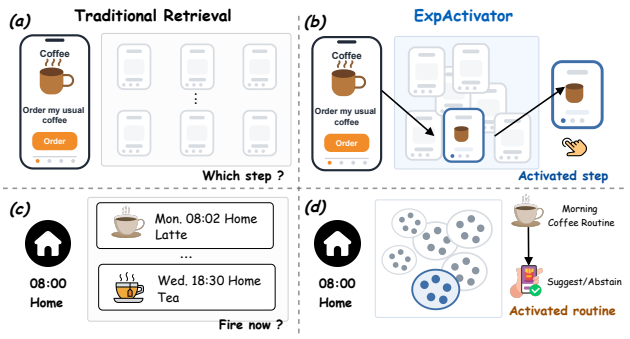


Figure 1: From retrieval to activation. Top: personalized execution. Bottom: proactive suggestion. (a) Retrieval supplies whole trajectories without indicating which step matches the current screen. (b) ExpActivator activates the matching historical step. (c) Retrieval returns related records without indicating whether a routine is due now. (d) ExpActivator activates the routine whose records match the current time and scenario, then suggests or abstains.

Our experiments reveal a sharp limitation of task-level retrieval. In execution, retrieved history strongly improves the opening decision, which commits the agent to an application, yet provides little sustained benefit over the remaining 90% of steps. This is because the instruction stays fixed while the screen changes at every step. A retrieved trajectory may indicate where to begin, but it does not tell the agent which past action fits *this* screen (Figure 1a). Proactive suggestion faces a similar problem. The same routine may be retrieved both at 8 a.m. at home and during an evening commute, even though it is appropriate only in the former (Chen et al., 2026; Wang et al., 2026; Nathani et al., 2026; Liu et al., 2026; Lin et al., 2026). Even methods that organize routines by time and scenario (Lyu et al., 2026) still leave the policy to decide whether the routine applies. Both cases stem from a gap between relevance and applicability. Retrieval identifies experience related to the task, whereas whether that experience applies depends on the situation at decision time.

We therefore recast personalization as *experience activation*: retrieval narrows history to relevant candidates, while the situation at each decision activates the experience that applies. We instantiate this principle in **ExpActivator**, a training-free framework operating at two granularities. During execution, the current screen activates the matching historical state in the frozen GUI backbone's latent space, and its associated action is supplied as a reference; the match is recomputed as the interface changes. Before execution, the current time and scenario determine whether a recurring intent is sufficiently supported, and the agent abstains otherwise (Figure 1b,d). Personalization thus becomes a decision-time selection problem rather than a one-time retrieval problem.

Across four GUI backbones, ExpActivator improves within-trajectory step success by 28% on average, achieves about 20% higher overall step success than the strongest baseline while using about one-fifth as many history tokens, and reaches approximately 2.3× its Matthews correlation coefficient for proactive suggestion. Controlled comparisons show that the same history helps substantially more when its applicable action is selected than when it is appended in full. Experience pays where it is activated, not where it is appended. Our contributions are as follows:

- **A diagnosis of task-level retrieval.** We show that task-level relevance does not imply decision-level applicability. Retrieved history helps mainly at the opening step of an episode and offers weak guidance on when a routine is due.
- **Experience activation at two levels.** We introduce ExpActivator, a training-free framework with step-level and task-level activation. Step-level activation matches every new screen to a historical state and uses its action as a reference, while task-level activation suggests a recurring intent only when the current time and scenario support it.
- **Evidence for decision-time activation.** Across four GUI backbones, step-level activation consistently improves personalized execution and task-level activation improves proactive suggestion, while using substantially less history context than existing approaches.

2 Method

2.1 Problem Formulation

For user u, let $\mathcal{M}_u = \{r_i\}_{i \in \mathcal{I}_u}$ denote the interaction history. Each record $r_i = (g_i, p_i, c_i, \tau_i, \xi_i)$ contains a goal, application, scenario, timestamp, and trajectory. The trajectory $\xi_i =$

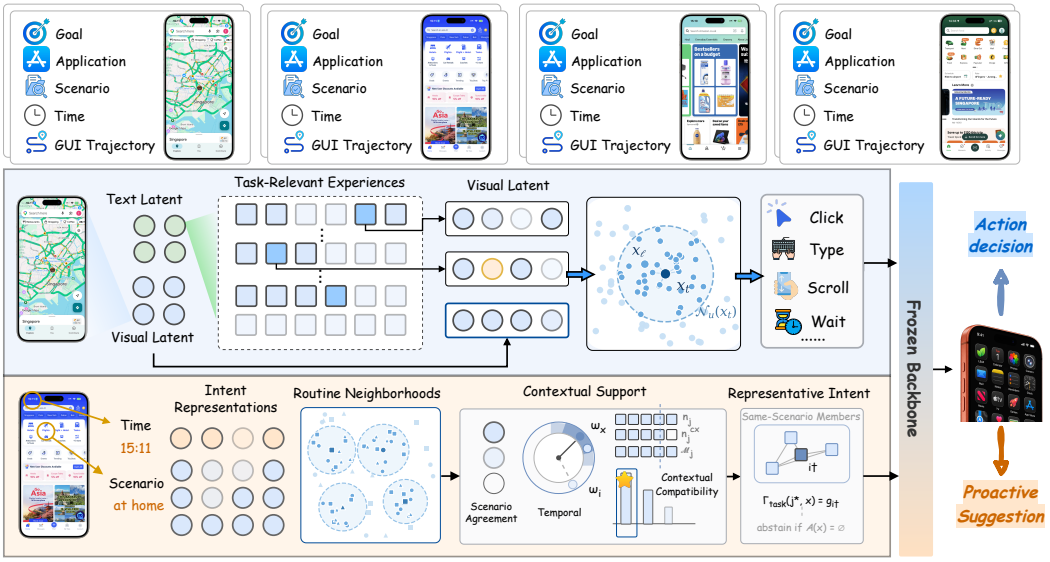


Figure 2: Overview of ExpActivator. Step-level activation (middle) matches each screen to recorded states of task-relevant trajectories and supplies the best-matching action as a reference. Task-level activation (bottom) scores recurring intents by scenario and temporal support and supplies a representative intent, or abstains. A frozen backbone outputs an action or suggestion.

$\{(s_{i,j}, a_{i,j})\}_{j=0}^{T_i-1}$ pairs each screen with the action taken from it. We consider two settings: instruction-driven execution, which predicts an action given a goal q, current screen s_t, and preceding actions $\mathcal{H}_t$; and proactive suggestion, which proposes a routine or abstains given a scenario and timestamp (c_x, τ_x). ExpActivator addresses both settings through *experience activation*: selecting historical behavior for the current decision rather than reusing an entire experience as a fixed script. Activation operates at two granularities. Step-level activation selects a historical action for execution, whereas task-level activation selects a historical intent for proactive suggestion. For $d \in \{\text{step}, \text{task}\}$, let $x = (q, s_t)$ in the former case and $x = (c_x, \tau_x)$ in the latter. For a nonempty candidate set, we define

$$e_u^{(d)}(x) = \Gamma_d\Big(\underset{b \in \mathcal{B}_u^{(d)}(x)}{\arg\max}\ \kappa_d(x, b),\ x\Big). \tag{1}$$

Here, $\mathcal{B}_u^{(d)}(x)$ indexes the candidate historical steps or routines, κ_d measures their compatibility with the current input, and Γ_d returns the associated action or intent. Frozen latent representations establish the semantic and visual relationships used for activation; time and scenario remain explicit contextual variables. Figure 2 traces both instantiations, which Sections 2.2 and 2.3 define in turn.

2.2 Step-Level Activation

Given q, ExpActivator first restricts retrieval to task-relevant experiences (Figure 2, middle). With a unit-normalized text encoder f, record relevance is $\alpha_i(q) = f(q)^\top f(\rho(g_i))$, where ρ removes the leading application phrase while retaining task content. We select $\mathcal{C}_u(q) = \mathrm{TopK}_{i \in \mathcal{I}_u^\xi;\, K_\xi}\, \alpha_i(q)$, where $\mathcal{I}_u^\xi$ contains records with readable trajectories. Before the first observation the same records also supply an initial reference e_0; its construction is routine. For subsequent observations $x_t = (q, s_t)$, the candidates are $\mathcal{B}_u^{(\text{step})}(x_t) = \{(i, j) : i \in \mathcal{C}_u(q),\ 0 \le j < T_i\}$. To compare the current screen with these historical states, ExpActivator uses the GUI backbone's visual features. Let $v_k(s)$ be a unit-normalized pooled token block and μ_k its mean over historical calibration images. The screen representation is

$$\phi(s) = \frac{1}{\sqrt{B}}\left[\nu(v_1(s) - \mu_1)\,; \ldots ; \nu(v_B(s) - \mu_B)\right], \tag{2}$$

where ν denotes ℓ_2 normalization. The B blocks follow a spatial layout, optionally augmented with a global block, selected through historical calibration for each backbone. The compatibility score $\kappa_{\text{step}}(x_t, (i,j)) = \phi(s_t)^\top \phi(s_{i,j})$ therefore averages block-wise cosine similarities between centered features. Equation 1 selects a historical step jointly over trajectories and their local states:

$$(i_t^*, j_t^*) = \arg \max_{\substack{i \in \mathcal{C}_u(q) \\ 0 \le j < T_i}} \phi(s_t)^\top \phi(s_{i,j}), \qquad e_t = \Gamma_{\text{step}}\big((i_t^*, j_t^*), x_t\big). \tag{3}$$

The operator Γ_{step} serializes the recorded action $a_{i_t^*, j_t^*}$ into the policy's reference format. While $\mathcal{C}_u(q)$ remains fixed within a task, the selected step is recomputed after every observation. The reference can therefore move between historical steps and trajectories without assuming that the current execution follows their step indices. The frozen policy generates $\widehat{a}_t = \pi_\theta(q, s_t, \mathcal{H}_t, e_t)$. The historical action serves as a reference, not a prescribed operation: the policy also observes the current screen to assess applicability and locate the target. State matching and policy inference repeat as execution progresses, allowing the referenced experience to change with the observed interface.

2.3 Task-level Activation

Without an explicit goal, ExpActivator identifies repeated tasks (Figure 2, bottom) from complete historical intents. Let $z_i = \psi(g_i)$ be the unit-normalized embedding from an intent encoder. Each center $j \in \mathcal{I}_u$ defines a routine neighborhood $\mathcal{R}_j(x) = \{i \in \mathcal{I}_u : \tau_i \neq \tau_x,\ z_j^\top z_i \ge \eta\}$, excluding records at the query timestamp. These neighborhoods group semantically similar occurrences and may overlap. A routine's support under the current context combines scenario agreement with temporal proximity:

$$M_j(x) = \sum_{i \in \mathcal{R}_j(x)} \mathbb{I}[c_i = c_x] \exp\left(-\frac{d_{24}(\omega_i, \omega_x)^2}{2\sigma^2}\right), \tag{4}$$

where $\omega_i = \text{hour}(\tau_i) \in [0, 24)$ and $d_{24}(a,b) = \min(|a-b|, 24 - |a-b|)$. Let $n_j = |\mathcal{R}_j(x)|$ denote total support and $n_j^{c_x} = \sum_{i \in \mathcal{R}_j(x)} \mathbb{I}[c_i = c_x]$ denote same-scenario support. ExpActivator distinguishes whether a routine has sufficient support for activation from how well it matches the present context:

$$\mathcal{A}(x) = \left\{ j \in \mathcal{I}_u : n_j \ge n_{\min},\ n_j^{c_x} \ge n_{\text{ctx}},\ M_j(x) \ge \delta \right\}, \tag{5}$$

$$\kappa_{\text{task}}(x, j) = \frac{M_j(x)}{n_j^\gamma}. \tag{6}$$

Thus, absolute support determines eligibility, while contextual match normalized by routine size determines ranking. We use $\gamma = 1$ (Table 5), so the ranking score is the mean contextual match per occurrence. Setting $\mathcal{B}_u^{(\text{task})}(x) = \mathcal{A}(x)$ instantiates the candidate set in Equation 1. If this set is empty, ExpActivator abstains; otherwise, it selects $j^* = \arg\max_{j \in \mathcal{A}(x)} \kappa_{\text{task}}(x, j)$. The activated intent is drawn from the selected routine's same-scenario members, $\mathcal{R}_{j^*}^{c_x} = \{i \in \mathcal{R}_{j^*}(x) : c_i = c_x\}$: we set $\Gamma_{\text{task}}(j^*, x) = g_{i^\dagger}$, where $i^\dagger = \arg\max_{i \in \mathcal{R}_{j^*}^{c_x}} \sum_{k \in \mathcal{R}_{j^*}(x)} z_i^\top z_k$ is the same-scenario member most similar, on average, to the entire routine. The corresponding historical intent is provided to the frozen backbone as a reference for suggestion generation, not as authorization to execute the task. All encoders and policy models remain frozen. Retrieval configurations and visual layouts are calibrated on held-out historical records and fixed during evaluation. Encoder specifications, calibration protocols, activation thresholds, and reference formats are detailed in Appendix B.

3 Experiment

3.1 Experimental Setup

Dataset. We evaluate on AndroidIntent (Lyu et al., 2026), built from long-term mobile records of 82 users across 135 applications and 13 scenario categories. The chronological first 80% of each user's records forms the history $\mathcal{M}_u$ (15,966 records), and the remaining episodes are evaluation targets excluded from retrieval. Personalized execution has 775 episodes with 7,915 steps whose instructions are deliberately ambiguous: a request such as checking updates names neither application nor content, so the intended behavior must be recovered from history. Proactive suggestion has 315 states (215 positive, 100 negative), each giving only a user, a time, and a scenario.

Table 1: Personalized execution and proactive suggestion on AndroidIntent. Within each block all methods share the same history and frozen policy. QwenVL-Max's BA and MCC are derived from its published recall and false-alarm rate, and Always suggest suggests on every state. Values are percentages, and bold marks the best result within each block.

		Personalized			Proactive			
Backbone	Method	Type↑	SSR↑	CSR↑	Sem.↑	P↑	BA↑	MCC↑
Qwen3-VL-8B	Base	47.6	19.4	23.7	–	–	–	–
	Recent	50.6	20.6	29.2	48.6	68.4	50.3	2.3
	Retrieve	51.6	22.6	41.4	48.4	67.8	48.9	−5.5
	LLM-UM	51.0	22.0	39.2	29.1	68.9	51.5	10.5
	HIM-Agent	51.3	22.6	40.2	50.8	72.7	58.5	20.7
	ExpActivator	**58.1**	**30.9**	**47.9**	**52.1**	**82.6**	**71.8**	**42.7**
MAI-UI-8B	Base	46.6	17.9	22.3	–	–	–	–
	Recent	47.6	18.1	21.2	44.5	68.3	50.0	0.0
	Retrieve	49.7	21.6	32.7	44.3	68.3	50.0	0.0
	LLM-UM	48.1	19.0	28.3	26.7	68.3	50.0	0.0
	HIM-Agent	48.2	19.9	28.7	**50.5**	70.6	54.8	14.1
	ExpActivator	**56.5**	**26.6**	**42.7**	49.6	**82.7**	**72.7**	**45.2**
Qwen3.5-9B	Base	50.1	17.2	21.4	–	–	–	–
	Recent	49.4	17.9	25.5	47.3	69.5	52.7	10.2
	Retrieve	49.2	19.9	32.5	**52.1**	68.2	50.0	0.0
	LLM-UM	52.4	20.7	39.8	21.5	61.9	49.0	−3.6
	HIM-Agent	51.4	21.0	37.8	48.4	82.4	61.2	22.0
	ExpActivator	**55.9**	**24.3**	**40.9**	51.0	**86.5**	**70.9**	**38.9**
GUI-Owl-1.5-8B	Base	58.2	22.7	21.5	–	–	–	–
	Recent	59.1	23.9	27.2	46.5	69.2	52.1	9.1
	Retrieve	60.3	28.2	41.6	48.2	67.8	49.1	−2.8
	LLM-UM	57.9	24.8	30.8	26.6	67.1	48.1	−3.9
	HIM-Agent	58.7	27.2	35.6	51.1	72.5	58.0	19.0
	ExpActivator	**61.8**	**28.9**	**45.8**	**53.9**	**82.6**	**72.4**	**44.6**
QwenVL-Max	–	51.6	24.8	27.3	52.2	67.4	49.6	−2.4
Always suggest		–	–	–	–	68.3	50.0	0.0

Metrics. For execution we report action-type accuracy (Type), step success rate (SSR), and the benchmark's cumulative success rate (CSR). For proactive suggestion we report the official semantic score of emitted suggestions (Sem.), precision (P), balanced accuracy (BA), and the Matthews correlation coefficient (MCC), which is zero for any rule that cannot tell positive from negative states; recall, false-alarm rate, and F1 are listed in Appendix C.5. Comparisons use paired bootstrap resampling, and all configurations are calibrated on held-out history before evaluation (Appendix B).

Backbones and baselines. We use four frozen open GUI backbones with different visual encoders and GUI training, Qwen3-VL-8B, MAI-UI-8B, Qwen3.5-9B, and GUI-Owl-1.5-8B, with the proprietary QwenVL-Max as an external reference. Within each backbone we compare ExpActivator with the history mechanisms of the benchmark: **Recent** supplies the latest records, **Retrieve** the records most similar to the instruction, **LLM-UM** an offline LLM-written profile, and **HIM-Agent** a hierarchical intent memory (Lyu et al., 2026); **Base** uses no history. All methods read the same history and share model revisions and inference settings, so comparisons are paired step by step.

3.2 Main Results

Personalized execution. Table 1 shows that ExpActivator achieves the best Type, SSR, and CSR on all four backbones, even though these backbones differ in visual encoder and GUI training. Relative to the same model without history, it improves SSR by 44% on average and roughly doubles CSR, and each of these gains is statistically significant under paired bootstrap. The margin over

Table 2: History-token cost and the two phases of an episode on Qwen3-VL-8B. History tokens are prompt tokens added over the no-history condition. Opening is each episode's first action and within-trajectory covers the 7,140 actions after it. The last column is the within-trajectory difference from Base with paired 95% user-cluster intervals. Rates are percentages.

	Prompt tokens/step		Step success rate			Within-trajectory
Method	History	Total	All	Opening	Within-traj.	− Base [95% CI]
Base	0.0	1011.4	19.39	28.26	18.43	–
Recent	756.6	1768.0	20.57	40.90	18.36	−0.07 [−0.87, +0.85]
Retrieve	730.6	1742.0	22.63	62.06	18.35	−0.08 [−1.16, +0.99]
LLM-UM	843.7	1855.1	22.01	62.32	17.63	−0.80 [−1.83, +0.12]
HIM-Agent	417.8	1429.2	22.58	62.32	18.26	−0.17 [−0.89, +0.57]
ExpActivator (ours)	**144.6**	**1156.0**	**30.92**	**70.06**	**26.67**	**+8.24** [+6.51, +10.16]

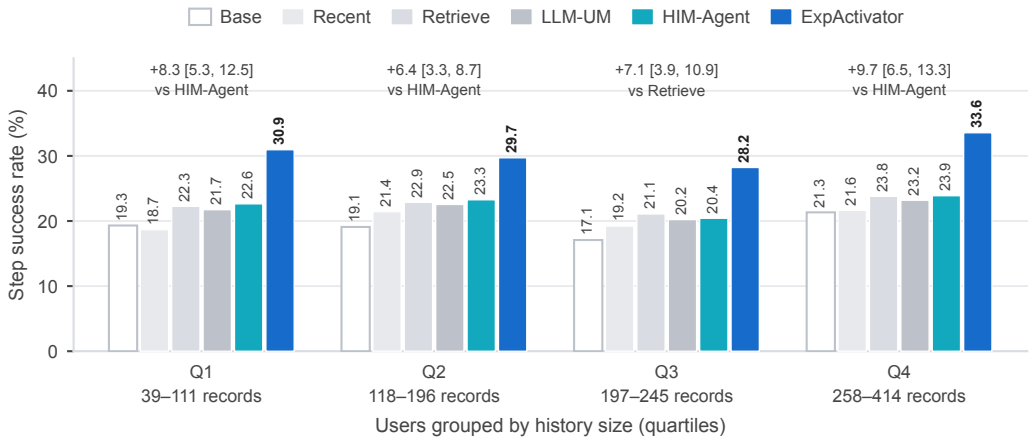


Figure 3: Step success by user history size on Qwen3-VL-8B. Users are grouped into quartiles by the records in their memory split, and all six conditions are scored on the same steps. Text above each group gives ExpActivator minus the strongest baseline there, with paired 95% user-cluster intervals.

the memory baselines is also substantial: ExpActivator improves SSR by about 20% on average over the strongest baseline in each block and by 37% on Qwen3-VL-8B, although every baseline reads exactly the same history. With ExpActivator, every open backbone surpasses the proprietary QwenVL-Max in CSR, reaching about 1.6 times its score on average. The improvement therefore comes neither from a larger policy nor from more history, but from supplying the part of the history that applies to the current step.

Proactive suggestion. A proactive assistant is useful only if it speaks at the right moments, and F1 cannot measure this on AndroidIntent: because 68% of states are positive, suggesting on every state already reaches 81.1% F1. We therefore evaluate the decision to suggest with BA and MCC, which credit a suggestion only when it separates the states that call for one from those that do not. By these measures the record- and profile-based baselines remain close to chance, and on MAI-UI three of them behave exactly like the always-suggest rule, since retrieving related records gives them no basis for staying silent. ExpActivator reaches about 2.3 times the MCC of the strongest baseline on average, improves BA by 24% relative to the best baseline, and attains the highest precision on every backbone, 84% on average. Deciding whether to speak before the backbone decides what to say is what turns accumulated routines into suggestions worth emitting.

3.3 Where Retrieved Experience Stops Paying

Because every method shares the backbone and the history, we can ask not only how much each one helps but for how long. An episode opens with one decision that commits it to an application and then proceeds through a trajectory of interface states, giving 775 openings and 7,140 subsequent

Table 3: Ablations with paired 95% user-cluster intervals. Step-level rows report policy SSR with one or eight retrieved trajectories held fixed; task-level rows report balanced accuracy at the declared operating point.

Control	Control score	ExpActivator	Gain [95% CI]
Step-level activation: policy SSR, $K_\xi = 1$			
Whole trajectory	18.47	23.84	5.36 [3.74, 7.06]
Step-level activation: policy SSR, $K_\xi = 8$			
Previous screen	22.63	26.67	4.03 [3.22, 4.84]
Update every four steps	23.17	26.67	3.50 [2.59, 4.47]
Task-level activation: balanced accuracy			
No time weighting	69.33	72.66	3.34 [0.15, 6.74]
No scenario condition	64.58	72.66	8.08 [2.64, 13.70]

actions. Scoring the two separately tests the question raised in Section 1: does a retrieved record remain usable once the trajectory is under way?

Retrieved experience is spent at the first decision. History matters most at the opening decision. On Qwen3-VL-8B, the strongest baselines raise it from 28.3% without history to 62.3%, and ExpActivator raises it to 70.1% (Table 2). Within the trajectory the picture reverses: over the remaining 7,140 actions, none of the four baselines is distinguishable from the no-history condition, and all stay within 5% of its success rate, even though they differ in what they retrieve and how they compress it. ExpActivator improves the same actions by 45% relative to the no-history condition. The pattern holds on every backbone, where no baseline adds more than 14% within the trajectory while ExpActivator adds 28% on average (Appendix F.2). Retrieval tells the agent which experience is related, and that answer is used up by the first decision; keeping experience useful afterwards requires selecting from it again at every observation.

Lower cost, from any amount of history. ExpActivator obtains these gains with a much lighter prompt. It supplies a single action reference per step and adds 144.6 history tokens, nearly five times fewer than the average baseline and nearly three times fewer than HIM-Agent, the most economical one, while scoring highest (Table 2; Figure 8 in Appendix F). No evaluated baseline therefore matches it on accuracy and cost at once. The benefit also does not depend on a long personal history. When users are grouped into quartiles by the size of their history, ExpActivator outperforms the strongest baseline in every group by more than 27% in relative SSR, and by 37% for the users with the fewest records (Figure 3). A modest number of past episodes already provides states that the current screen can activate.

3.4 Ablation Studies

Table 3 isolates each design choice, using the frozen Qwen3-VL-8B policy on the 7,140 within-trajectory actions and task-level activation on the 315 states; Appendix D gives the protocols.

Selecting beats appending. To isolate the value of selection, we restrict the history to the single top-ranked trajectory and compare two ways of using it. Supplying the trajectory whole, as retrieval-based memories do, yields 18.5% SSR, on par with using no history at all. Letting step-level activation select one action from the same trajectory raises SSR to 23.8%, a 29% relative improvement. The same experience therefore helps far more when the agent receives only the part that applies, which mirrors the within-trajectory failure of the memory baselines in Section 3.3.

Refresh at every observation. Selecting the right step also requires selecting it at the right moment. Holding the eight retrieved trajectories fixed, we keep the matching rule but feed it a stale screen. Matching against the previous screen lowers SSR by 15% relative, and refreshing the reference only every fourth step lowers it by 13%. Agreement between the selected reference and the recorded action drops even more sharply, by 38% relative, because a stale screen activates a step the user has already passed. Recomputing the match at every observation is therefore not a refinement of the mechanism but the mechanism itself.

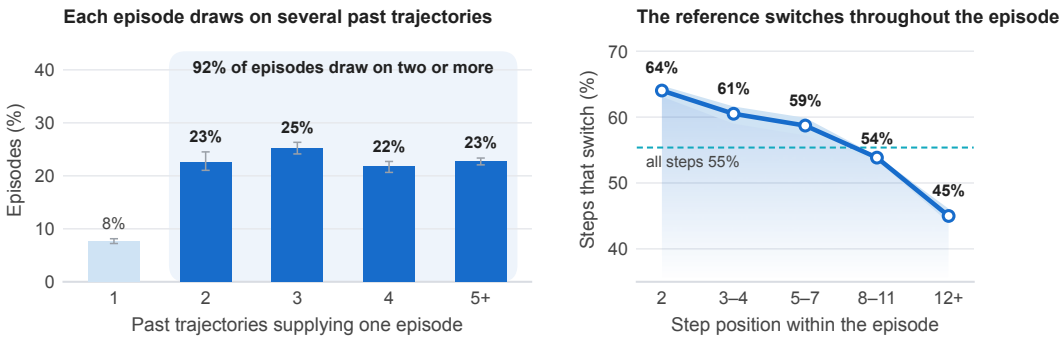


Figure 4: How step-level activation draws on past trajectories. Left: number of distinct past trajectories that supply the references of one episode. Right: share of steps whose reference comes from a different trajectory than at the previous step. Bars and the line give the mean over the four backbones; whiskers and the shaded band show their range.

Activation moves across trajectories. The refresh results imply that the activated reference should change as the interface changes, and Figure 4 shows that it does. On every backbone, about 92% of episodes draw their references from at least two past trajectories, 3.4 on average, and the reference moves to a different trajectory on 55% of consecutive steps. Switching is most frequent early in an episode, where generic screens resemble many past tasks, and it persists deep into the trajectory, still occurring on nearly half of the steps after the twelfth. These statistics are almost identical across the four backbones even though each matches screens with its own visual features, so the behavior follows from the activation rule rather than from any particular encoder. No single past trajectory, however relevant, could supply these references on its own.

Context gates the suggestion. Task-level activation relies on three contextual components, and removing each one hurts in a distinct way. Without time weighting, BA falls by about 5% relative, since repetition alone does not reveal when a routine applies. Without the scenario condition, BA falls by 11% relative and false alarms rise from 37% to 62%, so scenario agreement is what keeps activation selective. Frequency normalization leaves every trigger decision unchanged but improves the semantic quality of emitted suggestions by 4% relative, by favoring routines that match the context on each occurrence rather than on a few. Time decides when to fire, scenario decides whether to fire, and frequency decides what to say.

3.5 Case Study

Figure 5 shows both granularities of activation on real records. On the left, each current screen of a NUS NextBus task activates a recorded state that nearly duplicates it, and the reference moves from one past trip to another as the screens change, following the trip that matches the current stage of the task (Appendix G). On the right, the routine with the strongest contextual support for one user changes over the day, so task-level activation offers the weather at 08:05, plant identification at 15:37, the day's exercise at 20:26, and an alarm at 23:57, each matching what the user then did. Before dawn and shortly after 1 p.m., no routine has enough support and ExpActivator stays silent. Figures 11 and 12 show the same behavior for five more users.

4 Related Work

Personalized GUI Agents. Personalized GUI benchmarks evaluate whether agents follow user preferences across mobile tasks, with settings ranging from recorded demonstrations to interactive preference elicitation (Yang et al., 2025; Nie et al., 2026; Chen et al., 2026; Li et al., 2025). Methods use preference learning and hierarchical memory to recover information omitted from instructions (Wang et al., 2026; Lyu et al., 2026). The closest precedent, PersonalAlign, introduces AndroidIntent and HIM-Agent, which aggregates long-term records into preference and routine prototypes (Lyu et al., 2026). ExpActivator extends this focus on recovering user intent to selecting the

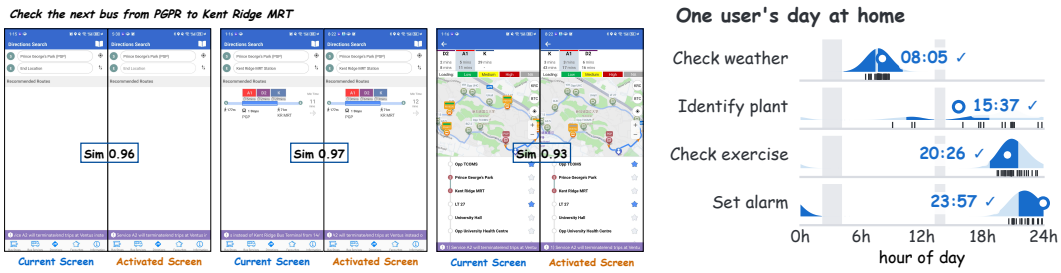


Figure 5: Activation on real records. Left: step-level activation places each current screen of one NUS NextBus task beside the recorded state it activates. Right: task-level activation for one user at home, showing four routines; each curve is a routine's contextual support over the day, dark segments mark when it is activated, gray bands mark abstention, and ticks mark its past occurrences. Intents are translated from Chinese.

historical behavior that can guide each decision. It distinguishes retrieving a task-relevant trajectory from identifying the historical step appropriate for the current screen.

Memory and Experience Reuse in GUI Agents. Reusable workflows and structured memories provide experience for GUI planning and action generation (Wang et al., 2024; Zhu et al., 2026; Qin et al., 2026; Chung et al., 2025; Chen et al., 2025). Memory can also be refreshed or selectively managed during execution (Zhu et al., 2026; Zeng et al., 2026; Liu et al., 2026). MementoGUI learns multimodal memory selection and compression with a frozen GUI backbone (Zeng et al., 2026), while ATMem tracks the role and status of task information and learns selective memory use (Liu et al., 2026). ExpActivator contributes a mechanism for selecting concrete actions from personal history without training a memory controller or the GUI policy. Step-level activation matches each screen against historical states within task-relevant trajectories and updates the action reference after every observation. The frozen policy combines this reference with the current screen and execution history to produce the action. Controlled comparisons with fixed candidate history support the value of screen matching and reference updates for personalized action prediction.

Proactive Assistance. Proactive mobile benchmarks study how user history, screen activity, and device context support intent inference without explicit requests (Yang et al., 2025; Chai et al., 2026; Kong et al., 2026). Interactive evaluations further examine intervention timing and responses to feedback (Chen et al., 2026; Nathani et al., 2026). Closest to our setting, HIM-Agent uses temporal and scenario regularities to organize routine memory and supplies representative intents and states for proactive reasoning (Lyu et al., 2026). ExpActivator uses this contextual information in an explicit activation rule applied before suggestion generation. Task-level activation separates the support required to consider a routine from the contextual score used to rank it, and abstains when no routine qualifies. ExpActivator provides a common basis for using personal history across execution and suggestion: current conditions select an action or intent reference, while the frozen backbone generates the response. Both thus adapt their use of experience without fine-tuning.

5 Conclusion

We have argued that personal experience helps only where it is activated: a relevant trajectory need not contain an action suited to the current screen, and a familiar routine may become due even without any instruction. ExpActivator acts on this principle without any training. Step-level activation re-anchors the action reference to every observation, while task-level activation fires only when time and scenario support a routine. The same record store then yields, across four frozen backbones, the best personalized execution and the only selective proactive suggestion among the evaluated methods, at the lowest history-token cost. For personal agents, the lesson is compact: experience pays where it is activated, not where it is appended.

Reproducibility Statement

ExpActivator is training-free, and all encoders and GUI backbones remain frozen. Section 2 specifies the activation rule (Eq. 1), step-level activation (Eqs. 2–3), and task-level activation (Eqs. 4–6). Section 3.1 specifies the chronological history split of AndroidIntent, the backbones, the baselines, and the evaluation metrics. Appendix A gives the notation and pseudocode for both granularities (Algorithms 1 and 2). Appendix B lists all hyperparameters and inference settings (Table 5), the calibration protocols, the reference formats, and the prompt templates. Appendix C gives the dataset statistics, evaluation protocol, metric definitions, paired user-level bootstrap, and baseline construction. Appendices D–F give the ablation protocols, the threshold sensitivity analysis, and the token accounting and history-size analyses. Code and per-example outputs will be released subject to the license and privacy terms of AndroidIntent.

AI Use Statement

Generative AI tools were used to help organize author-provided source material, search for related work, draft and edit prose, and write experiment and analysis scripts. The authors reviewed the resulting text, verified all reported claims and numerical results against the run outputs, and take full responsibility for the final manuscript.

References

Yuxiang Chai, Shunye Tang, Han Xiao, Rui Liu, and Hongsheng Li. Pira-bench: A transition from reactive gui agents to gui-based proactive intent recommendation agents. *arXiv preprint arXiv:2603.08013*, 2026.

Tongbo Chen, Zhengxi Lu, Zhan Xu, Guocheng Shao, Shaohan Zhao, Fei Tang, Yong Du, Kaitao Song, Yizhou Liu, Yuchen Yan, Wenqi Zhang, Xu Tan, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Knowu-bench: Towards interactive, proactive, and personalized mobile agent evaluation, 2026. URL https://arxiv.org/abs/2604.08455.

Yurun Chen, Xavier Hu, Yuhan Liu, Ziqi Wang, Zeyi Liao, Lin Chen, Feng Wei, Yuxi Qian, Bo Zheng, Keting Yin, et al. Graph2eval: Automatic multimodal task generation for agents via knowledge graphs. *arXiv preprint arXiv:2510.00507*, 2025.

Andy Chung, Yichi Zhang, Kaixiang Lin, Aditya Rawal, Qiaozi Gao, and Joyce Chai. Evaluating long-context reasoning in llm-based webagents, 2025. URL https://arxiv.org/abs/2512.04307.

Lawrence Keunho Jang, Andrew Keunwoo Jang, Jing Yu Koh, and Ruslan Salakhutdinov. Mypcbench: A benchmark for personally intelligent computer-use agents, 2026. URL https://arxiv.org/abs/2606.16748.

Serin Kim, Sangam Lee, and Dongha Lee. Persona2web: Benchmarking personalized web agents for contextual reasoning with user history, 2026. URL https://arxiv.org/abs/2602.17003.

Dezhi Kong, Zhengzhao Feng, Qiliang Liang, Hao Wang, Haofei Sun, Changpeng Yang, Yang Li, Peng Zhou, Shuai Nie, Hongzhen Wang, et al. Proactivemobile: A comprehensive benchmark for boosting proactive intelligence on mobile devices. *arXiv preprint arXiv:2602.21858*, 2026.

Ning Li, Qiqiang Lin, Zheng Wu, Xiaoyun Mo, Weiming Zhang, Yin Zhao, Xiangmou Qu, Jiamu Zhou, Jun Wang, Congmin Zheng, Yuanyi Song, Hongjiang Chen, Heyuan Huang, Jihong Wang, Jiaxin Yin, Jingwei Yu, Junwei Liao, Qiuying Peng, Xingyu Lou, Jun Wang, Weiwen Liu, Zhuosheng Zhang, and Weinan Zhang. Coloragent: Building a robust, personalized, and interactive os agent, 2025. URL https://arxiv.org/abs/2510.19386.

Wanli Li, Bowen Zhou, Yunyao Yu, Zhou Xu, Yifan Yang, Dongsheng Li, and Caihua Shan. Weavebench: A long-horizon, real-world benchmark for computer-use agents with hybrid interfaces, 2026. URL https://arxiv.org/abs/2606.09426.

Zhixin Lin, Jungang Li, Dongliang Xu, Shidong Pan, Yibo Shi, Yuchi Liu, Yuecong Min, and Yue Yao. Mobile gui agent privacy personalization with trajectory induced preference optimization, 2026. URL https://arxiv.org/abs/2604.11259.

Chen Liu, Ling Chen, Hanzhang Zhou, Xu Zhang, Quyu Kong, Panrong Tong, Wenhao Wang, Xin Yu, Steven Hoi, and Yue Wang. What memory do GUI agents really need? from passive records to active task-driving states. *arXiv preprint arXiv:2606.31612*, 2026. URL https://arxiv.org/abs/2606.31612.

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. *arXiv preprint arXiv:2601.09636*, 2026.

Deepak Nathani, Cheng Zhang, Chang Huan, Jiaming Shan, Yinfei Yang, Alkesh Patel, Zhe Gan, William Yang Wang, Michael Saxon, and Xin Eric Wang. Proactive agent research environment: Simulating active users to evaluate proactive assistants. *arXiv preprint arXiv:2604.00842*, 2026.

Hongyi Nie, Xunyuan Liu, Yudong Bai, Yaqing Wang, Yang Liu, Quanming Yao, and Zhen Wang. PSPA-Bench: A personalized benchmark for smartphone GUI agent. *arXiv preprint arXiv:2603.29318*, 2026. URL https://arxiv.org/abs/2603.29318.

Zerui Qin, Sheng Yue, Xingyuan Hua, Yongjian Fu, and Ju Ren. Executable agentic memory for GUI agent. *arXiv preprint arXiv:2605.12294*, 2026. URL https://arxiv.org/abs/2605.12294.

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, 2026.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024. URL https://arxiv.org/abs/2409.07429.

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. *arXiv preprint arXiv:2507.21071*, 2025.

Zhe Yang, Shulin Tian, Kairui Hu, Shuai Liu, Hoang-Nhat Nguyen, Yichi Zhang, Zujin Guo, Mengying Yu, Zinan Zhang, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Hippocamp: Benchmarking contextual agents on personal computers, 2026. URL https://arxiv.org/abs/2604.01221.

Mengqi Yuan, Zilong Zhou, Xinzhuang Xiong, Weiming Wu, Jiayang Sun, Jiamin Song, Kaiqian Cui, Bowen Wang, Haoyuan Wu, Yitong Li, Dunjie Lu, Haikong Lu, Qi Zhen, Xinyuan Wang, Jiaqi Deng, Yuhao Yang, Cheng Chen, Boyuan Zheng, Alex Su, Xiao Yu, Hao Zou, Saaket Agashe, Xing Han Lu, Manpreet Kaur, Zhengyang Qi, Vincent Sunn Chen, Frederic Sala, Dayiheng Liu, Junyang Lin, Zhou Yu, Yu Su, Siva Reddy, Xin Eric Wang, Peng Qi, Tianbao Xie, and Tao Yu. Osworld 2.0: Benchmarking computer use agents on long-horizon real-world tasks, 2026. URL https://arxiv.org/abs/2606.29537.

Ziyun Zeng, Hang Hua, Bocheng Zou, Mu Cai, Rogerio Feris, and Jiebo Luo. MementoGUI: Learning agentic multimodal memory control for long-horizon GUI agents. *arXiv preprint arXiv:2605.18652*, 2026. URL https://arxiv.org/abs/2605.18652.

Sibo Zhu, Wenyi Wu, Kun Zhou, Stephen Wang, and Biwei Huang. Hybrid self-evolving structured memory for GUI agents. *arXiv preprint arXiv:2603.10291*, 2026. URL https://arxiv.org/abs/2603.10291.

APPENDIX

A Notation and Algorithms **13**

A.1 Notation . . . 13

A.2 Algorithms . . . 13

B Implementation Details **14**

B.1 Hyperparameters and Inference Settings . . . 14

B.2 Step-level Activation: Application Choice . . . 14

B.3 Step-level Activation: Screen Representation . . . 15

B.4 Step-level Activation: Action Reference . . . 15

B.5 Task-level Activation: Intent Reference . . . 15

B.6 Prompt Templates . . . 16

C Benchmark and Evaluation Protocol **16**

C.1 Dataset Statistics . . . 16

C.2 Data and Evaluation Protocol . . . 17

C.3 Metric Definitions . . . 17

C.4 Baseline Construction . . . 18

C.5 Complete Proactive Metrics . . . 18

D Ablation Protocols and Additional Results **19**

D.1 Step-level Activation: Selecting versus Appending . . . 19

D.2 Step-level Activation: Reference Updates . . . 19

D.3 Task-level Activation: Time, Scenario, and Frequency . . . 20

E Threshold Sensitivity of Task-level Activation **20**

F Context Cost and History-Size Protocols **21**

F.1 Token accounting . . . 21

F.2 Within-Trajectory Gains on All Backbones . . . 22

F.3 History-size stratification . . . 23

G Case Study **24**

G.1 Task-level Activation for More Users . . . 24

A Notation and Algorithms

A.1 Notation

Table 4 collects the symbols used in Section 2.

Table 4: Notation.

Symbol	Meaning
$\mathcal{M}_u,\ \mathcal{I}_u$	interaction history of user u and its record indices
$r_i = (g_i, p_i, c_i, \tau_i, \xi_i)$	record: goal, application, scenario, timestamp, trajectory
$\xi_i = \{(s_{i,j}, a_{i,j})\}_{j<T_i}$	recorded screens and the actions taken from them
$q,\ s_t,\ \mathcal{H}_t$	instruction, current screen, preceding actions
(c_x, τ_x)	scenario and timestamp of a proactive query
$\mathcal{B}_u^{(d)}(x),\ \kappa_d,\ \Gamma_d$	candidates, compatibility, and readout at granularity d
$f,\ \alpha_i(q),\ K_\xi,\ \mathcal{C}_u(q)$	goal encoder, record relevance, number and set of candidate trajectories
$\phi(s),\ B,\ \mu_k$	centered block screen representation, number of blocks, block means
$(i_t^*, j_t^*),\ e_t$	activated historical step and its action reference at step t
$\psi,\ z_i,\ \eta$	intent encoder, intent embedding, routine threshold
$\mathcal{R}_j(x),\ n_j,\ n_j^{c_x}$	routine neighborhood, total and same-scenario support
$M_j(x),\ \sigma$	contextual mass and temporal bandwidth
$n_{\min},\ n_{\mathrm{ctx}},\ \delta,\ \gamma$	support, scenario, mass thresholds and frequency exponent
$j^*,\ i^\dagger$	activated routine and its representative intent

A.2 Algorithms

Algorithms 1 and 2 give the two instantiations of Equation 1. Neither updates a parameter: every quantity is either precomputed once from the history with frozen encoders or computed at inference from the current input.

Algorithm 1 Step-level activation for personalized execution

Require: instruction q; history $\mathcal{M}_u$; frozen goal encoder f, screen representation ϕ, policy π_θ; budget K_ξ

Offline: cache $f(\rho(g_i))$ for every record and $\phi(s_{i,j})$ for every recorded screen

$\mathcal{C}_u(q) \leftarrow \mathrm{TopK}_{i \in \mathcal{I}_u^\xi;\, K_\xi}\ f(q)^\top f(\rho(g_i))$ ▷ fixed for the whole task

$e_0 \leftarrow$ initial reference from the same records (Appendix B.2); $\mathcal{H}_0 \leftarrow \emptyset$

for $t = 0, 1, 2, \ldots$ **do**

  observe the current screen s_t

  if $t > 0$ **then**

    $(i_t^*, j_t^*) \leftarrow \arg\max_{i \in \mathcal{C}_u(q),\, 0 \le j < T_i} \phi(s_t)^\top \phi(s_{i,j})$ ▷ joint search over trajectories and states

    $e_t \leftarrow \Gamma_{\mathrm{step}}\big((i_t^*, j_t^*), x_t\big)$ ▷ serialize $a_{i_t^*, j_t^*}$

  end if

  $\widehat{a}_t \leftarrow \pi_\theta(q, s_t, \mathcal{H}_t, e_t)$ ▷ the policy may adopt, adapt, or reject e_t

  if $\widehat{a}_t$ terminates the task **then**

    break

  end if

  execute $\widehat{a}_t$; $\mathcal{H}_{t+1} \leftarrow \mathcal{H}_t \cup \{\widehat{a}_t\}$

end for

Cost. Both procedures are training free, and their offline work is linear in the history: each historical screen passes once through the backbone's visual tower and each goal once through the text encoder. Online, step-level activation adds one visual encoding of s_t and $\sum_{i \in \mathcal{C}_u(q)} T_i$ inner products of dimension $B \cdot d$ per observation, a cost that is independent of the total history size because $\mathcal{C}_u(q)$ holds at most $K_\xi = 8$ trajectories. The policy is still called once per step, as in Base, and reads a single short reference rather than whole records (Section 3.3). Task-level activation scans the precomputed similarity matrix of one user, with at most 414 records in this benchmark, before the backbone verbalizes the activated intent.

Algorithm 2 Task-level activation for proactive suggestion

Require: query (c_x, τ_x); history $\mathcal{M}_u$; frozen intent encoder ψ; frozen backbone; $\eta, \sigma, n_{\min}, n_{\mathrm{ctx}}, \delta, \gamma$

Offline: cache $z_i = \psi(g_i)$ and the similarity matrix $[z_j^\top z_i]_{j,i \in \mathcal{I}_u}$
for each center $j \in \mathcal{I}_u$ **do**
  $\mathcal{R}_j(x) \leftarrow \{i : \tau_i \neq \tau_x,\ z_j^\top z_i \geq \eta\}$; $n_j \leftarrow |\mathcal{R}_j(x)|$; $n_j^{c_x} \leftarrow \sum_{i \in \mathcal{R}_j(x)} \mathbb{I}[c_i = c_x]$
  $M_j(x) \leftarrow \sum_{i \in \mathcal{R}_j(x)} \mathbb{I}[c_i = c_x]\ \exp\!\big(-d_{24}(\omega_i, \omega_x)^2 / 2\sigma^2\big)$
end for
$\mathcal{A}(x) \leftarrow \{j : n_j \geq n_{\min},\ n_j^{c_x} \geq n_{\mathrm{ctx}},\ M_j(x) \geq \delta\}$ ▷ eligibility by absolute support
if $\mathcal{A}(x) = \emptyset$ **then**
  return ABSTAIN
end if
$j^* \leftarrow \arg\max_{j \in \mathcal{A}(x)} M_j(x) / n_j^\gamma$ ▷ ranking by match per occurrence
$i^\dagger \leftarrow \arg\max_{i \in \mathcal{R}_{j^*}(x),\, c_i = c_x} \sum_{k \in \mathcal{R}_{j^*}(x)} z_i^\top z_k$
return suggestion verbalized by the frozen backbone from the intent reference $g_{i^\dagger}$

B Implementation Details

B.1 Hyperparameters and Inference Settings

Table 5 lists every setting used in the reported runs. All values are fixed before evaluation and shared by the four backbones unless stated otherwise.

Table 5: Hyperparameters and inference settings.

Component	Setting	Value
Step-level	goal encoder f	paraphrase-multilingual-MiniLM-L12-v2
	candidate trajectories K_ξ	8
	screen layout B	4 centered quadrants; MAI-UI adds 1 global block
	reference coordinate frame	normalized to $[0, 999]$
Task-level	intent encoder ψ	paraphrase-multilingual-MiniLM-L12-v2
	routine threshold η	0.94
	supports $n_{\min}$, n_{ctx}	12, 4
	bandwidth σ, mass δ, exponent γ	1 hour, 0.25, 1
Inference	decoding	greedy (temperature 0), seed 0
	response and context length	256 and 8,192 tokens
	screenshot budget	200,704 pixels, one image per request
	precision and engine	BF16, vLLM 0.28.0
	hardware	one GPU per run
Evaluation	suggestion encoder	Qwen3-Embedding-0.6B
	bootstrap	20,000 paired draws over users

B.2 Step-level Activation: Application Choice

The goal encoder f in Section 2.2 is `paraphrase-multilingual-MiniLM-L12-v2`. Goal preprocessing removes only the leading application-opening phrase; the remaining instruction is kept as the retrieval key. The canonical application label p_i is compiled once from the historical package name and complete instruction using frozen Qwen3-VL-8B-Instruct. The same frozen MiniLM encoder embeds complete intents for routine formation. The official suggestion similarity metric uses Qwen3-Embedding-0.6B. Visual descriptor normalization is $\nu(v) = v / \max(\|v\|_2, \epsilon)$ with $\epsilon = 10^{-12}$.

For the initial reference e_0 in step-level activation, let π_r index the r-th most similar same-user record according to $\alpha_i(q)$. The application reference is obtained by

$$V(p \mid q, u) = \sum_{r=1}^{K_a} \mathbb{I}[p_{\pi_r} = p] \, \frac{[\alpha_{\pi_r}(q)]_+}{r}, \qquad p^* = \arg\max_p V(p \mid q, u), \\ e_0 = \texttt{open_app}(p^*). \tag{7}$$

where $[a]_+ = \max(a, 0)$. Lexical order resolves exact application-vote ties. The application reference e_0 is supplied through the same frozen-policy interface as the action references selected at later steps. We use $K_a = 10$ and $K_\xi = 8$, selected on 3,226 historical application queries and 1,015 actions from 83 pseudo-held-out trajectories, respectively.

B.3 Step-level Activation: Screen Representation

Each visual block is the normalized mean of a specified set of tokens from the GUI backbone under evaluation. Candidate blocks include global means from visual layers 8, 16, 24, and the final layer, as well as means over the four quadrants of the final spatial token grid. The spatial layout uses the four quadrant blocks; the pyramid layout adds the final-layer global block. Intermediate blocks provide additional calibration candidates. Their readout uses the backbone's native visual features; Qwen3.5 uses unmerged visual tokens. Centering statistics μ_k are estimated from the distinct historical reference images in the calibration retrieval pools and then reused during evaluation.

For each user with sufficient readable history, the final readable historical trajectory provides calibration queries and the earlier trajectories provide reference candidates. All descriptor alternatives are compared on the same 932 within-trajectory actions for each backbone. Calibration evaluates both raw normalized blocks and centered blocks. Calibration selects four centered spatial blocks for Qwen3-VL, Qwen3.5, and GUI-Owl, and four centered spatial blocks plus one global block for MAI-UI. These screen representations use Eq. 2, with $B = 4$ for the spatial layout and $B = 5$ for the pyramid layout.

In Qwen3-VL, Qwen3.5, GUI-Owl, and MAI-UI order, historical action-matching accuracy is 35.8%, 35.4%, 35.1%, and 36.6%, compared with 29.6% for pixel mean squared error (RGB-MSE). Exact McNemar p-values are 1.5×10^{-4}, 1.8×10^{-4}, 5.1×10^{-4}, and 2.1×10^{-5}, respectively. These calibration results determine the descriptor used for the subsequent policy evaluation.

B.4 Step-level Activation: Action Reference

The operator Γ_{step} in Equation 3 preserves the historical action type and its structured arguments. For a historical click at pixel coordinates (x, y) on a screen of size (W, H), the action reference uses

$$(x^e, y^e) = \left(\text{round}\left(999 \frac{x}{W}\right), \text{round}\left(999 \frac{y}{H}\right) \right). \tag{8}$$

The same convention applies to long clicks. Scroll direction and typed text retain their structured fields; wait, back, home, and termination preserve their action types. These fields describe the historical reference. The policy receives the current screenshot, instruction, action prefix, and the action reference and predicts the current operation as described in Section 2.2.

B.5 Task-level Activation: Intent Reference

For task-level activation (Section 2.3), circular time distance in Eq. 4 is $d_{24}(a, b) = \min(|a-b|, 24-|a-b|)$ for hours in $[0, 24)$. The routine threshold is $\eta = 0.94$, minimum total support is $n_{\min} = 12$, minimum scenario support is $n_{\text{ctx}} = 4$, temporal bandwidth is $\sigma = 1$ hour, minimum weighted support is $\delta = 0.25$, and frequency normalization exponent is $\gamma = 1$. The exclusion $\tau_i \neq \tau_x$ applies to neighborhood membership; each historical intent remains eligible to serve as a neighborhood center.

After selecting j^*, ExpActivator identifies a representative among the members sharing the query scenario. Its centrality is computed against every member of the selected semantic routine:

$$i^\dagger = \arg \max_{\substack{i \in \mathcal{R}_{j^*}(x) \\ c_i = c_x}} \sum_{k \in \mathcal{R}_{j^*}(x)} z_i^\top z_k, \qquad e_u^{(\text{task})}(x) = g_{i^\dagger}. \tag{9}$$

Source index resolves centrality ties within numerical tolerance 10^{-7}. The task-level activation ablations score this historical intent directly, before suggestion generation. In the backbone comparison, the selected intent is supplied as an intent reference and the frozen backbone verbalizes the suggestion. Both evaluations use the same task-level activation rules.

B.6 Prompt Templates

Step-level activation reaches the policy as one structured field appended to the backbone's native execution prompt; the system prompt, action space, and output format are those of Base. The user turn reads:

```
## User Profile
{user_profile}
## Task Progress
{history_actions}
## User's Query
{instruction}
## Grounded Historical Evidence
{"evidence_type": "action_reference",
 "retrieved_action": {"action": "click", "coordinate": [183, 599]},
 "historical_coordinate_frame": "0-999 normalized",
 "retrieval_basis": "same-user residual-goal trajectory retrieval and
                     visual state matching"}

The evidence is a candidate next action retrieved from personal history; its
retrieval_basis field states whether it comes from a goal-level vote or a
goal-and-screen match. Use it to guide the current decision, but do not copy it mechanically.
Inspect the current screenshot and task progress, then adopt, adapt, or
reject the candidate as needed. For a click or long press, locate the
target on the current screenshot and produce current coordinates.
```

For task-level activation, the activated routine is passed to the prompt below, which HIM-Agent uses for its own routines. The routine is serialized as JSON with the fields shown; when ExpActivator abstains, the field is an empty list.

```
You are skilled at analyzing user history. You are given summarized user
daily routines. Determine whether the current user state requires a
proactive suggestion.
## Note
- Each summarized routine describes a frequent behavior with intent,
  time/scenario distribution, and frequency.
- If neither the time nor the scenario strongly matches a summarized
  routine, output False.
- If a recommendation is needed, output a suitable user instruction for
  the GUI agent to execute.
- Express the user's intent unambiguously in one Chinese sentence without
  adding extra description.
- Do not output explanations, time, or scenario information. Output only
  the user instruction or False.
## Input
User_profile: {profile}   Time: {time}   Scenario: {scenario}
Summarized_routine: [{"routine_intent": g, "historical_state": {time, scenario},
  "occurrence_count": n_j, "matched_scenario_count": n_j^c,
  "temporal_state_mass": M_j, "normalized_state_likelihood": M_j / n_j}]
The user's intent:
```

C Benchmark and Evaluation Protocol

C.1 Dataset Statistics

Table 6 summarizes the evaluation data. Histories vary by an order of magnitude across users, and most target steps are taken on an interface that the instruction does not describe: clicks and scrolls alone account for 59% of all steps.

Table 6: Statistics of the history and evaluation splits. Action shares are over the 7,915 execution steps.

History and execution		Action type share (%)	
Users	82	click	41.1
History records	15,966	scroll	17.9
Records per user (median)	196.5	wait	13.6
Records per user (min, max)	39, 414	open app	9.8
Execution episodes	775	finish	9.8
Execution steps	7,915	type	5.8
Steps per episode (mean, median)	10.2, 8	back, home, recent	1.8
Steps per episode (max)	85	long click	0.3
Proactive states (positive, negative)	215, 100		

C.2 Data and Evaluation Protocol

The chronological first 80% of each user's records form the historical pool. Every target execution episode is excluded from execution retrieval. Proactive queries contain user identity, time, and scenario; records with the exact query timestamp are excluded from routine membership as in Section 2.3. Historical calibration fixes retrieval neighborhood sizes, descriptor layouts, and centering statistics before policy evaluation. The semantic encoders, visual towers, application-label compiler, and GUI policies remain frozen.

$\mathcal{H}_t$ in Section 2.2 is the benchmark's preceding action sequence. Reference selection uses the current instruction and screen, while the current target action is reserved for scoring. Appendix D details the step-level activation experiments on selecting versus appending and on reference updates, followed by the task-level activation experiments on time, scenario, and frequency normalization.

C.3 Metric Definitions

Execution. For a predicted action $\widehat{a}_t$ and the recorded action a_t, type accuracy counts $\widehat{a}_t$ as correct when the action types agree. Step success additionally requires the arguments to agree: a click or long click must fall within a normalized distance of 0.14 of the target on a 1000×1000 grid, a scroll must share the direction, typed text must reach a character similarity of 0.7, and an application opening must name the target application; back, home, wait, and finish match on type alone. SSR averages step success over all steps. The benchmark's cumulative success rate weights the steps of an episode of length L by

$$\mathrm{CSR} = \frac{\sum_e \sum_{k<L_e} w_k^{(L_e)}\, m_{e,k}}{\sum_e \sum_{k<L_e} w_k^{(L_e)}}, \qquad w_k^{(L)} = L\,\frac{e^{-0.6k}}{\sum_{k'<L} e^{-0.6k'}}, \tag{10}$$

where $m_{e,k} \in \{0,1\}$ is step success at position k of episode e. CSR therefore emphasizes the early decisions that determine whether a personalized episode starts on the right path.

Proactive suggestion. A method emits a suggestion or abstains on each state. With true and false positives and negatives counted over the 315 states, precision is $P = \mathrm{TP}/(\mathrm{TP}+\mathrm{FP})$, recall is $R = \mathrm{TP}/(\mathrm{TP}+\mathrm{FN})$, the false-alarm rate is $\mathrm{FA} = \mathrm{FP}/(\mathrm{FP}+\mathrm{TN})$, and $\mathrm{BA} = \frac{1}{2}(R + 1 - \mathrm{FA})$. The Matthews correlation coefficient is

$$\mathrm{MCC} = \frac{\mathrm{TP}\cdot\mathrm{TN} - \mathrm{FP}\cdot\mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}, \tag{11}$$

which is zero for a rule that suggests on every state and, unlike F1, is not inflated by the 68% share of positive states. The semantic score follows the benchmark: for every suggestion emitted on a positive state it averages the normalized edit similarity and the Qwen3-Embedding-0.6B cosine similarity between the suggestion and the recorded intent, and Sem. reports the mean over these suggestions.

Statistical comparison. Every comparison is paired, since all methods are evaluated on the same steps or states. We resample users with replacement, keep all of a resampled user's steps or states

together, and recompute the metric difference on each of 20,000 draws; a difference is called significant when its 95% percentile interval excludes zero.

C.4 Baseline Construction

All baselines read the same chronological history as ExpActivator and are inserted through the native prompt of each backbone. **Recent** supplies the latest three records with their intent, application, and actions. **Retrieve** supplies the three same-user records whose intents are most similar to the instruction under Qwen3-Embedding-0.6B. **HIM-Agent** uses the released history filter, intent clustering with a minimum cluster size of three, and top-three retrieval with the released record format. **LLM-UM** supplies an offline per-user profile of frequent behaviors with example actions; since the original profiles are not released, we generate them deterministically with a local Qwen3.5-9B from each user's most recent 150 history records. For proactive suggestion, HIM-Agent and ExpActivator share one prompt that presents summarized routines, while Recent, Retrieve, and LLM-UM share one that presents raw history records. Every context was fixed for all states and episodes before inference.

C.5 Complete Proactive Metrics

Table 7 lists the benchmark's recall, false-alarm rate, and F1 alongside precision and the suggestion score. Because 68% of the 315 states are positive, a rule that suggests on every state attains 81.1 F1 with no ability to tell states apart, which is why the main table summarizes the activation decision with balanced accuracy and the Matthews correlation coefficient.

Table 7: Complete proactive metrics on 215 positive and 100 negative states. All values are percentages; lower is better for FA.

Backbone	Method	Sem.	P	R	FA	F1
Qwen3-VL-8B	Recent	48.6	68.4	98.6	98.0	80.8
	Retrieve	48.4	67.8	95.8	98.0	79.4
	LLM-UM	29.1	68.9	99.1	96.0	81.3
	HIM-Agent	50.8	72.7	87.9	71.0	79.6
	ExpActivator	52.1	82.6	79.5	36.0	81.0
MAI-UI-8B	Recent	44.5	68.3	100.0	100.0	81.1
	Retrieve	44.3	68.3	100.0	100.0	81.1
	LLM-UM	26.7	68.3	100.0	100.0	81.1
	HIM-Agent	50.5	70.6	91.6	82.0	79.8
	ExpActivator	49.6	82.7	82.3	37.0	82.5
Qwen3.5-9B	Recent	47.3	69.5	95.3	90.0	80.4
	Retrieve	52.1	68.2	34.0	34.0	45.3
	LLM-UM	21.5	61.9	6.0	8.0	11.0
	HIM-Agent	48.4	82.4	41.4	19.0	55.1
	ExpActivator	51.0	86.5	62.8	21.0	72.8
GUI-Owl-1.5-8B	Recent	46.5	69.2	96.3	92.0	80.5
	Retrieve	48.2	67.8	90.2	92.0	77.4
	LLM-UM	26.6	67.1	70.2	74.0	68.6
	HIM-Agent	51.1	72.5	86.0	70.0	78.7
	ExpActivator	53.9	82.6	81.9	37.0	82.2
Always suggest		–	68.3	100.0	100.0	81.1

Table 8: Qwen3-VL-8B controls on the 7,140 within-trajectory actions. Whole trajectory supplies the entire record. Previous historical action replaces the selected action with the action immediately before it in the historical trajectory.

Action reference	Policy SSR
Step-level activation, $K_\xi = 1$	
Whole trajectory	18.47
Selected reference	23.84
Step-level activation, $K_\xi = 8$	
Previous historical action	23.32
Selected reference (ExpActivator)	26.67

D Ablation Protocols and Additional Results

D.1 Step-level Activation: Selecting versus Appending

Protocol. We use the existing history split, retrieval scores, and history-calibrated visual representation without additional parameter tuning. For each test episode we fix the highest-ranked usable trajectory retrieved from that user's historical pool. The Whole trajectory condition supplies this trajectory as a complete record, and the Selected reference condition applies step-level activation (Section 2.2) to the same trajectory and supplies one action. Both run the frozen Qwen3-VL-8B policy on the 7,140 within-trajectory actions with the current screen and the ground-truth preceding actions visible. A second control, at $K_\xi = 8$, replaces the selected reference with the action immediately preceding it in the same historical trajectory. Intervals resample the 82 users with replacement 20,000 times, and inference uses greedy decoding with seed 0, a maximum context length of 8,192 tokens, and up to 256 output tokens.

Results. Selecting one reference raises policy SSR from 18.47% to 23.84%, a gain of 5.36 [3.74, 7.06] points. At $K_\xi = 8$, the preceding historical action reaches 23.32% against 26.67% for the selected reference, a gain of 3.35 [2.34, 4.46] points. Table 8 collects both controls.

D.2 Step-level Activation: Reference Updates

The reference-update experiment uses the frozen Qwen3-VL-8B checkpoint with a 200,704-pixel image budget, 256-token output cap, 8,192-token context limit, 256-request batches, and at most 64 concurrent sequences. The policy receives the same user profile, ambiguous query, current screen, and action prefix. The reference-update experiment changes only the action reference supplied by step-level activation. At within-trajectory position t, Previous screen uses $s_{\max(1,t-1)}$; Update every four steps uses $s_{1+4\lfloor(t-1)/4\rfloor}$. Both leave the reference at the first step unchanged. The retrieved trajectories stay fixed, so the action reference for a stored observation can be reused exactly.

Table 9: Step-level activation with eight fixed historical trajectories. All conditions run the frozen Qwen3-VL-8B policy on the same current inputs and 7,140 within-trajectory actions. Rates are percentages; differences are Current screen minus control in percentage points with 95% paired user-cluster intervals.

Action reference	Policy SSR	Difference [95% CI]
Previous screen	22.63	4.03 [3.22, 4.84]
Update every four steps	23.17	3.50 [2.59, 4.47]
Current screen (ExpActivator)	26.67	–

D.3 Task-level Activation: Time, Scenario, and Frequency

Task-level activation ablations use the same frozen MiniLM index over 15,966 historical intents and expose only user identity, time and scenario to selection. The full condition reproduces every production suggestion and source record. No time weighting sets the exponential factor in Equation 4 to one. No scenario condition removes the scenario indicator, the minimum same-scenario support requirement, and the same-scenario restriction on the selected intent. No frequency normalization ranks routines by $M_j(x)$ instead of $M_j(x)/n_j^\gamma$ in Equation 6, while keeping the eligible set unchanged. Activation curves vary the threshold on the maximum support $M_j(x)$, computed under each condition, among routines meeting the minimum occurrence requirements, with a score of -1 for queries without such a routine. These are ROC diagnostics of the declared score, including standard endpoints; dots identify the actual selector's fixed-threshold decisions. The official Semantic scorer uses Qwen3-Embedding-0.6B and fuzzy edit similarity.

Table 10: Task-level activation before suggestion generation on 215 positive and 100 negative states. BA is balanced accuracy and FA is false-alarm rate. Semantic is the official score on emitted positive suggestions. All conditions use the same activation threshold. All values are percentages.

Condition	AUROC	BA	Recall	FA	Semantic
ExpActivator	80.37	72.66	82.33	37.00	55.35
No time weighting	74.10	69.33	84.65	46.00	50.61
No scenario condition	79.95	64.58	91.16	62.00	52.67
No frequency normalization	80.37	72.66	82.33	37.00	53.24

At the declared operating point, balanced accuracy falls by 3.34 [0.15, 6.74] points without time weighting and by 8.08 [2.64, 13.70] points without the scenario condition, and removing time weighting also lowers AUROC by 6.28 [1.15, 11.96] points. Frequency normalization raises Semantic on emitted suggestions by 2.11 [0.61, 4.10] points without changing any trigger decision.

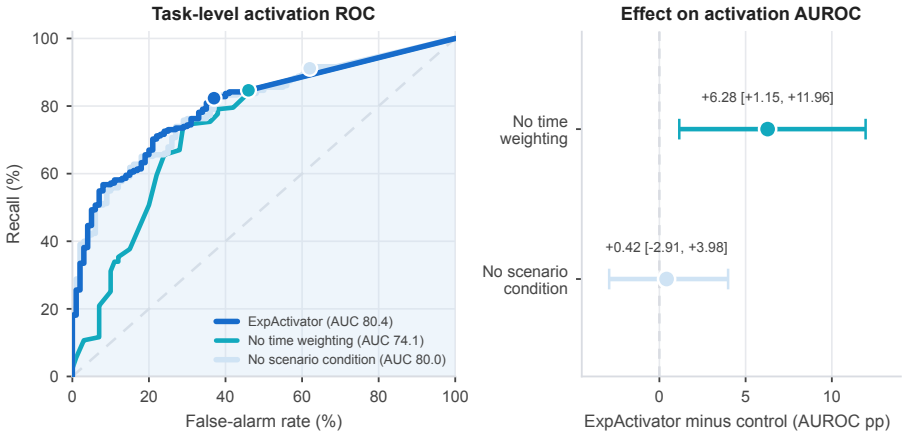


Figure 6: Task-level activation ROC curves and paired AUROC differences. Dots mark the original threshold. Error bars are 95% user-cluster bootstrap intervals. No frequency normalization has the same trigger decisions and curve as ExpActivator. The curves include the standard ROC endpoints described above.

E Threshold Sensitivity of Task-level Activation

Task-level activation declares six constants: the routine threshold η, the minimum total support $n_{\min}$, the minimum same-scenario support n_{ctx}, the temporal bandwidth σ, the activation threshold δ, and the normalization exponent γ. This section re-runs the activation rule of Section 2.3 over the sealed routine index for grids of these constants, so the reported operating point can be read against the

surface it sits on. Only activation is re-run: no suggestion is generated and no model is called. The reimplementation reproduces the sealed run exactly at the declared setting (AUROC 80.37, balanced accuracy 72.66, recall 82.33, false alarms 37.00), which is asserted before any sweep. The sweeps are diagnostic: every reported result uses the declared constants.

Routine threshold and minimum support. Over $\eta \in [0.88, 0.98]$ and $n_{\min} \leq 12$, activation AUROC stays above 75 in all 40 cells and varies by less than 5.4 points (Figure 7); the declared setting $(\eta, n_{\min}) = (0.94, 12)$ reaches 80.37, within 0.33 points of the grid maximum, so the operating point sits on a broad plateau rather than a sharp peak. Beyond $n_{\min} = 12$ the surface falls away, to 73.24 at $n_{\min} = 16$ and 58.95 at $n_{\min} = 24$. This is a coverage effect rather than an instability of the score: requiring that many repetitions leaves progressively more states with no eligible routine at all, rising from 27.6% at $n_{\min} = 12$ to 41.9% at 16 and 73.0% at 24, and those states receive the constant abstention score, which removes the discrimination the metric measures. The usable range of $n_{\min}$ is therefore bounded above by how often the user's routines actually repeat, not by a tuned threshold.

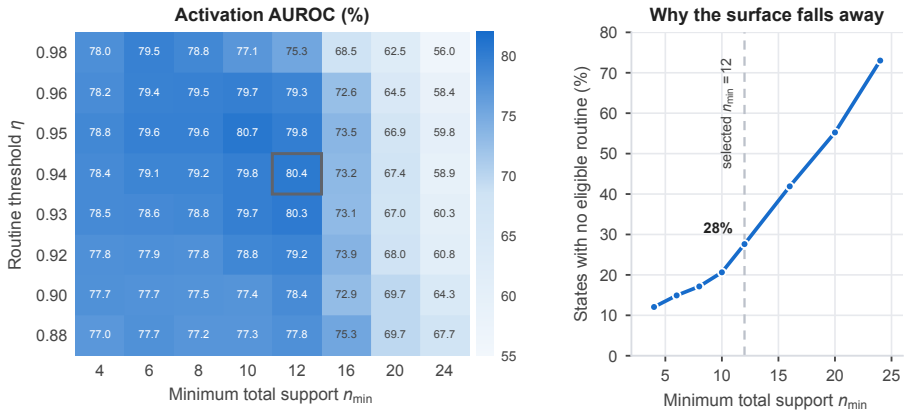


Figure 7: Sensitivity of task-level activation. Left: activation AUROC over the routine threshold η and the minimum total support $n_{\min}$, with the declared setting outlined. Right: the share of the 315 states for which no routine satisfies the support requirements, at $\eta = 0.94$. The AUROC decline beyond $n_{\min} = 12$ tracks this loss of coverage.

Bandwidth, scenario support, and activation threshold. Table 11 sweeps the temporal bandwidth σ and the same-scenario support n_{ctx} one at a time. Temporal bandwidth is flat around its declared value: AUROC changes by less than 0.6 points for $\sigma \in [0.25, 2]$ hours and declines only once the kernel spans several hours, when time stops distinguishing routines. Same-scenario support moves AUROC by less than 1.8 points over $n_{\text{ctx}} \in [1, 10]$, while larger values steadily lower the false-alarm rate. The activation threshold δ cannot change AUROC, because it selects an operating point on a fixed ranking; over $\delta \in [0.05, 1.0]$ balanced accuracy changes by less than 1.3 points. The ROC curve of ExpActivator in Figure 6 traces every operating point that δ can select.

Frequency normalization. The exponent γ ranks routines inside the eligible set and cannot move an activation decision. Setting $\gamma \in \{0, 0.5, 1.5\}$ leaves every trigger decision identical to the declared $\gamma = 1$ while changing the selected routine on 100, 41, and 44 of the emitted states, respectively. Its effect is therefore confined to which intent is verbalized, which Appendix D.3 scores with the official suggestion metric.

F Context Cost and History-Size Protocols

F.1 Token accounting

Every prompt is rebuilt with the evaluator's own message assembly, chat template, and tokenizer, using the checkpoint revision that produced the run, and is then tokenized without inference. The

Table 11: One-at-a-time sensitivity of task-level activation on 215 positive and 100 negative states, with the other constants held at their declared values. BA is balanced accuracy, FA is the false-alarm rate, and No routine is the share of states for which no routine meets the support requirements. Declared settings are marked †. All values are percentages except the swept constant.

Temporal bandwidth σ (hours)						Same-scenario support n_{ctx}					
σ	AUROC	BA	Recall	FA	No routine	n_{ctx}	AUROC	BA	Recall	FA	No routine
0.25	79.92	73.05	72.09	26.00	27.6	1	79.82	69.59	84.19	45.00	21.6
0.5	79.96	73.07	78.14	32.00	27.6	2	79.77	69.90	82.79	43.00	24.4
0.75	80.20	72.73	80.47	35.00	27.6	3	80.07	71.66	82.33	39.00	27.0
1†	80.37	72.66	82.33	37.00	27.6	4†	80.37	72.66	82.33	37.00	27.6
1.5	80.51	71.40	82.79	40.00	27.6	5	79.74	71.50	80.00	37.00	28.9
2	80.26	70.90	82.79	41.00	27.6	6	79.62	71.50	80.00	37.00	29.2
3	79.24	70.86	83.72	42.00	27.6	8	78.94	71.57	78.14	35.00	32.4
4	78.11	69.09	84.19	46.00	27.6	10	78.59	71.44	74.88	32.00	34.9
6	76.98	69.33	84.65	46.00	27.6						

screenshot is supplied to the policy as a separate visual input that is byte-identical across conditions, so the text-token difference from Base isolates the history each mechanism adds. The serialized action reference is counted exactly as it is sent, including a constant field naming the retrieval basis, so the reported cost of ExpActivator is an overestimate. Base is the same no-history condition reported in Table 1. The opening and within-trajectory groups partition the evaluation by position in the episode, which in this benchmark coincides with a partition by kind of decision: the first recorded action of every episode commits it to an application, and every later action is taken on an observed interface state.

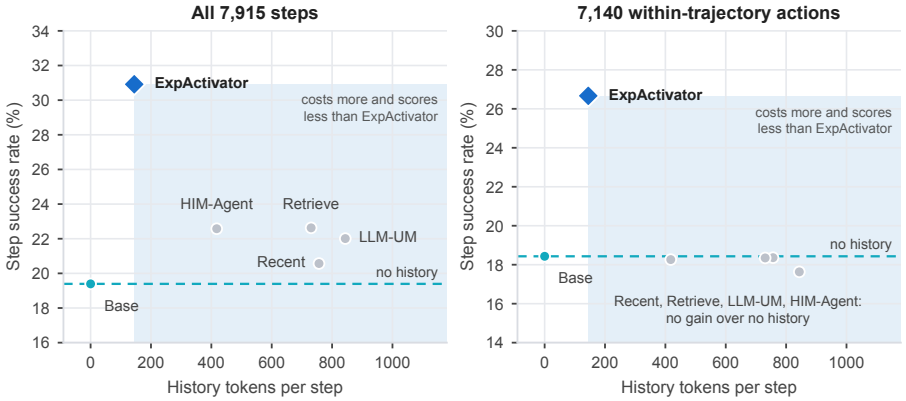


Figure 8: Step success against history tokens per step on Qwen3-VL-8B. The shaded region holds every point that costs more and scores less than ExpActivator. Left: all steps. Right: actions within the trajectory, where the four memory baselines stay at the no-history level.

F.2 Within-Trajectory Gains on All Backbones

Table 12 repeats the within-trajectory comparison of Table 2 on every backbone. Across the sixteen pairings of a memory baseline with a backbone, the gain over using no history never exceeds 2.4 points and is indistinguishable from zero in twelve, while ExpActivator improves every backbone.

Table 12: Within-trajectory step success over the 7,140 actions after each episode's first decision. Base is the no-history rate in percent; other columns give the difference from Base in percentage points with paired 95% user-cluster bootstrap intervals (20,000 draws).

Backbone	Base	Recent	Retrieve	LLM-UM	HIM-Agent	ExpActivator
Qwen3-VL-8B	18.43	−0.07 [−0.87, +0.85]	−0.08 [−1.16, +0.99]	−0.80 [−1.83, +0.12]	−0.17 [−0.89, +0.57]	+8.24 [+6.51, +10.16]
MAI-UI-8B	17.41	+0.77 [−0.07, +1.63]	+2.34 [+1.32, +3.36]	+0.41 [−0.39, +1.27]	+0.98 [+0.21, +1.75]	+5.67 [+4.73, +6.69]
Qwen3.5-9B	16.57	−0.20 [−1.04, +0.73]	+0.88 [−0.27, +2.11]	−0.10 [−1.05, +0.86]	+0.56 [−0.37, +1.50]	+3.74 [+2.79, +4.76]
GUI-Owl-1.5-8B	22.30	+0.48 [−0.59, +1.58]	+1.72 [+0.75, +2.72]	+0.63 [−0.17, +1.42]	+1.92 [+1.24, +2.64]	+2.38 [+0.53, +4.41]

F.3 HISTORY-SIZE STRATIFICATION

ExpActivator exceeds the strongest baseline within every quartile of history size by at least 6.4 points, with all four intervals excluding zero (Figure 9); within the trajectory, the strongest comparator is the no-history condition itself in three of the four groups. Users in the smallest quartile, with at most 111 records, gain 8.31 [5.32, 12.50] points, so step-level activation is usable from a small personal history.

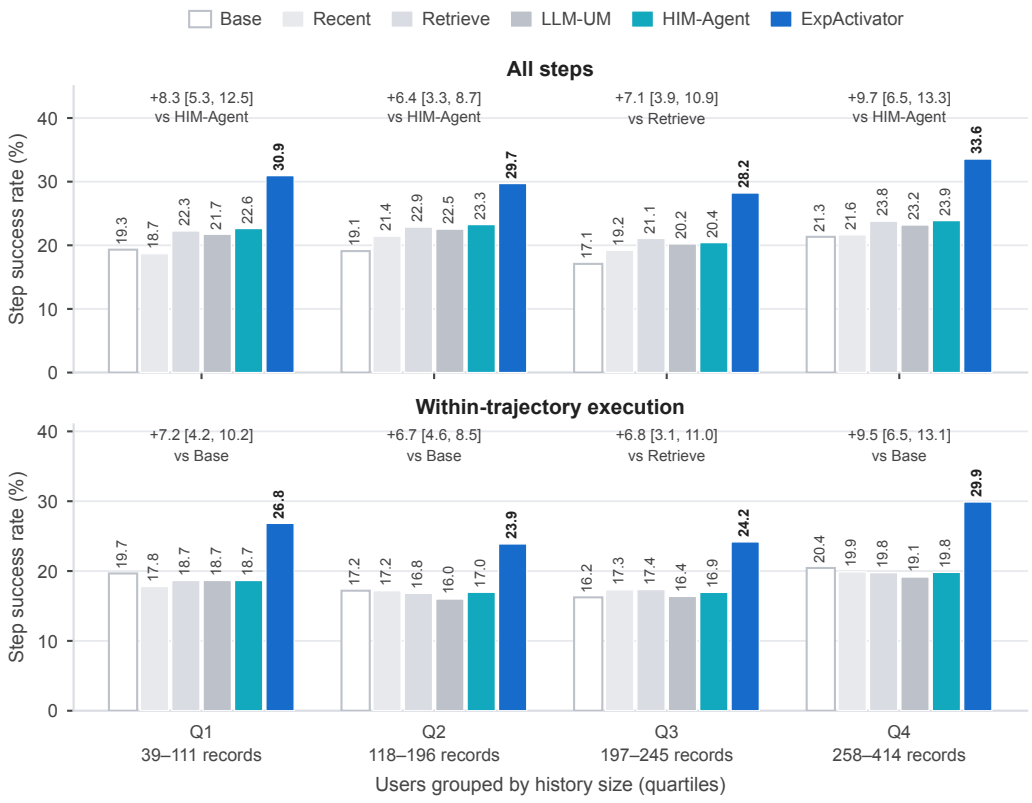


Figure 9: Personalized execution by history size on Qwen3-VL-8B. Users are grouped into quartiles by the number of records in their memory split, and all six conditions are scored on the same steps inside each quartile. Top: all steps. Bottom: within-trajectory execution, where the four memory baselines stay at the no-history level in every quartile. Blue text gives ExpActivator minus the strongest baseline in that group, in percentage points with paired 95% user-cluster bootstrap intervals; the strongest baseline is selected per group and is named, so the comparator changes across groups.

User history size is the number of records in that user's chronological first-80% memory split, the same pool all methods retrieve from. The 82 evaluated users are divided into quartiles by this count; quartile boundaries come from the history alone and use no evaluation output. Within each quartile every condition is scored on the same steps, and intervals resample whole users with replacement 20,000 times, as elsewhere in this paper. The strongest baseline is chosen separately inside each quartile, so the comparator is not fixed across buckets and the reported gain is the smallest available against any baseline there. Quartiles contain unequal step counts because episode length and the number of test episodes vary across users.

Within the trajectory the strongest comparator is the no-history condition in three of the four quartiles, consistent with the aggregate decomposition in Table 2. Both analyses are reproduced by `analyze_ExpActivator_context_cost.py` and `analyze_`

Table 13: Personalized execution by user history size on Qwen3-VL-8B. Users are grouped into quartiles by records in their memory split. Rates are step success percentages on the steps belonging to each group. Gain is ExpActivator minus the strongest baseline within that group, in percentage points with paired 95% user-cluster bootstrap intervals.

Group	Users	Records	Steps	Base	Recent	Retrieve	LLM-UM	HIM-Agent	ExpActivator	Strongest	Gain [95% CI]
All steps											
Q1	21	39–111	782	19.31	18.67	22.25	21.74	22.63	30.95	HIM-Agent	+8.31 [+5.32, +12.50]
Q2	20	118–196	1535	19.09	21.43	22.87	22.54	23.26	29.71	HIM-Agent	+6.45 [+3.29, +8.74]
Q3	20	197–245	2424	17.08	19.22	21.08	20.21	20.42	28.22	Retrieve	+7.14 [+3.90, +10.86]
Q4	21	258–414	3174	21.33	21.64	23.79	23.19	23.88	33.55	HIM-Agent	+9.67 [+6.48, +13.25]
All	82	39–414	7915	19.39	20.57	22.63	22.01	22.58	30.92	Retrieve	+8.29 [+6.37, +10.40]
Within-trajectory execution											
Q1	21	39–111	697	19.66	17.79	18.65	18.65	18.65	26.83	Base	+7.17 [+4.24, +10.25]
Q2	20	118–196	1344	17.19	17.19	16.82	16.00	16.96	23.88	Base	+6.70 [+4.59, +8.46]
Q3	20	197–245	2242	16.24	17.31	17.35	16.37	16.95	24.17	Retrieve	+6.82 [+3.07, +10.96]
Q4	21	258–414	2857	20.44	19.88	19.78	19.15	19.81	29.89	Base	+9.45 [+6.47, +13.05]
All	82	39–414	7140	18.43	18.36	18.35	17.63	18.26	26.67	Base	+8.24 [+6.51, +10.16]

`ExpActivator_history_size.py` from the sealed run files; neither performs additional model inference.

G Case Study

Figure 10 traces step-level activation on a real task. The references for steps 2 and 6 come from an April 9 trip that leaves the same residence for AS3, because those screens match the current ones. Once the route screens for Kent Ridge MRT appear, the reference moves to an April 11 trip to that destination, and the frozen policy selects route A1 and ends the task.

G.1 Task-level Activation for More Users

Figures 11 and 12 repeat Figure 5 (right) for five more users, with every routine that is activated at some hour shown. In every case the activated routine follows the time of day, and for the commuter it also follows the scenario: two minutes apart, the same user is offered the e-bus card on the street and a study task once on board. Every marked state activates the routine the user performed, and every user has hours in which no routine qualifies and ExpActivator abstains.

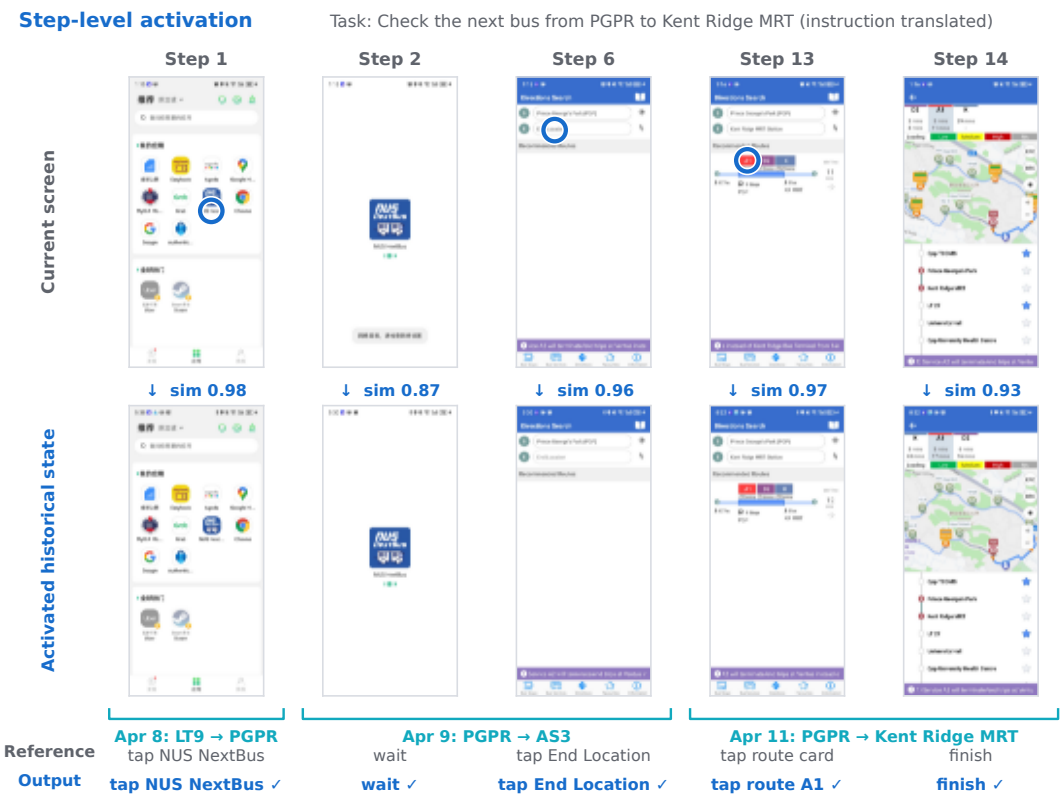


Figure 10: Step-level activation on five steps of one NUS NextBus task. Each current screen activates the most similar recorded state among eight task-relevant trajectories, so the reference moves from one past trip to another as the screen changes; circles mark the frozen policy's taps. The task instruction is translated from Chinese.

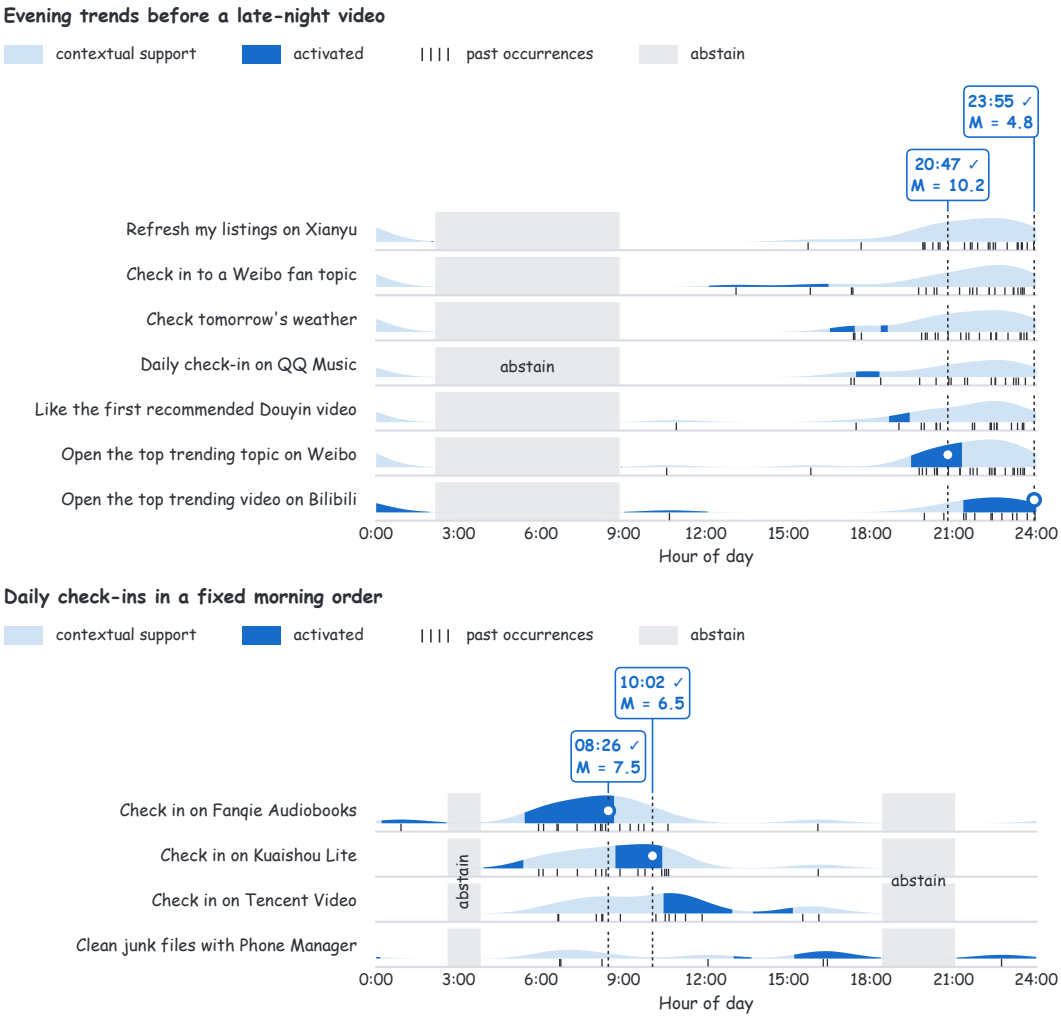


Figure 11: Task-level activation for two more users, in the format of Figure 5 (right). Intents are translated from Chinese.

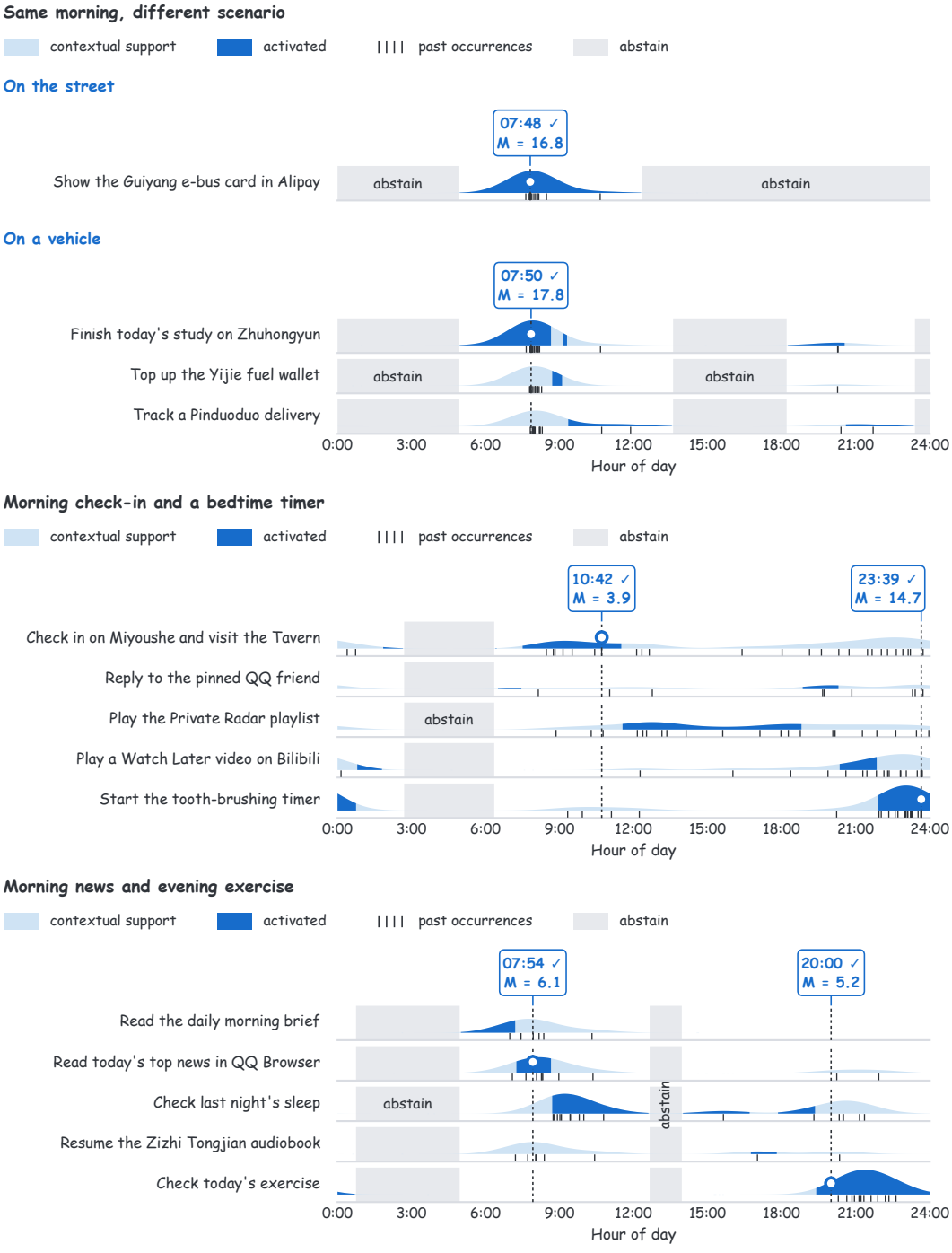


Figure 12: Task-level activation for three more users, in the format of Figure 5 (right). Top: one commuter on the street and on a vehicle on the same morning. Intents are translated from Chinese.