COMPULSIONBENCH: A BENCHMARK FOR AUDITING COMPULSION-RISK PROXIES IN SEQUENTIAL RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce COMPULSIONBENCH, a reproducible benchmark for auditing and mitigating compulsion-risk proxies in sequential recommendation under explicit, non-clinical assumptions. COMPULSIONBENCH asks when engagement-oriented policies interact with long-horizon user dynamics to increase observable behavioral proxies, including prolonged sessions, late-night session starts, rapid-return diagnostics, and watch time accumulated beyond a fixed session threshold. It also asks which mitigation strategies reduce those proxies under matched evaluation. Simulator-internal quantities, such as continuation despite a latent reflective preference to stop, are reported only as post-hoc mechanism diagnostics and are not benchmark targets. COMPULSIONBENCH combines a partially observed user simulator, tunable platform controls for personalization and reward variability, mitigation interventions including session caps and observable-risk constrained optimization, a public-log calibration protocol with released aggregate audit artifacts, and a standardized scorecard that separates observable compulsion-risk proxies from latent mechanism diagnostics. The released benchmark provides two plug-compatible user-response backends: a transparent parametric simulator used as the reference environment, and an LLM-augmented semantic robustness backend that preserves the same observation space, action space, interventions, calibration targets, and scorecard while perturbing the item-response kernel under the same benchmark interface. COMPULSIONBENCH defines two benchmark tasks, engagement maximization and observable-risk constrained optimization, and reports tail-risk and mitigation metrics for every policy. In the released active-threshold profile, high-watch deterministic heuristics produce higher over-cap exposure than PPO, while a simple session cap reduces PPO over-cap exposure from 2.84 to 0.45 minutes at a 4.1% watch-time cost. Episode-level diagnostics further show that the cap is a tail-severity intervention: over-cap incidence is essentially unchanged, but deterministic p95 over-cap exposure falls from 14.23 to 2.36 minutes without a short-gap re-entry spike. A reference 120-minute profile shows that the same over-cap channel can be inactive, highlighting the need to report threshold provenance and proxy activation. COMPULSIONBENCH makes audits of compulsion-risk proxies and mitigation strategies reproducible while remaining explicit about construct-validity limits.

1 INTRODUCTION

Modern content platforms rely on recommendation policies that optimize observable proxies such as click-through rate, dwell time, or watch time. These signals are useful for learning and easy to measure at scale, but they are not reliable stand-ins for what users value, would endorse on reflection, or would prefer in the long run (Ekstrand & Willemsen, 2016; Milli et al., 2021; Kleinberg et al., 2024; Agarwal et al., 2024; Anwar et al., 2025). Observed engagement can conflate interactions that create lasting utility with interactions driven primarily by short-run impulse or temptation (Agarwal et al., 2024; Anwar et al., 2025). This creates a central challenge for recommender systems: a policy can improve engagement while moving the system away from outcomes users would endorse on reflection (Milli et al., 2021; Kleinberg et al., 2024).

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.

The challenge becomes sharper in sequential, feed-based settings. Recommendation policies do not merely predict immediate responses; they shape what users see next, how preferences evolve, when users return, and how difficult it becomes to disengage (Kalimeris et al., 2021; Dean & Morgenstern, 2022; Curmei et al., 2022). Long-horizon optimization is also increasingly central in recommender systems, including objectives related to retention, repeated visits, and long-term engagement (Zou et al., 2019; Xue et al., 2023; Liu et al., 2024). In such environments, the relevant failure modes are long-horizon: unusually prolonged sessions, rapid re-entry after stopping, elevated late-night use, or continued accumulation of watch time beyond a long-session threshold. These are benchmarkable *compulsion-risk proxies*: they arise from sequential exposure, accumulate over time, and can be amplified or reduced by policy choices.

Despite rapid progress in recommender simulation and RL evaluation, the field lacks a standard benchmark centered on auditing those proxies directly. Existing infrastructure, including RecSim, RecSim NG, KuaiSim, and EasyRL4Rec, has made it easier to model multi-step recommendation, represent uncertainty, and compare sequential decision-making algorithms (Ie et al., 2019; Mladenov et al., 2021; Zhao et al., 2023; Yu et al., 2024). Public short-video datasets such as KuaiRand and KuaiRec make session-level analysis and calibration of synthetic environments more practical (Gao et al., 2022b;a). However, existing simulators and benchmark suites are not organized around a common task of auditing engagement-driven policies for compulsion-risk proxies under explicit construct-validity limits. Recent work on value-aware, utility-aware, and temptation-aware recommendation makes the gap sharper: if engagement is not the same as value, the field needs evaluation settings that make that mismatch measurable (Milli et al., 2021; Agarwal et al., 2024; Anwar et al., 2025).

CompulsionBench makes conditional policy audits reproducible: under stated assumptions about user dynamics and platform actions, when do engagement-oriented policies increase observable risk proxies, and how much can mitigations reduce them? Its tasks and scorecard separate these observable outcomes from latent diagnostics.

The shared interface supports two aligned backends: CompulsionBench-Param, a closed-form reference simulator, and CompulsionBench-LLM, an optional semantic perturbation of its response kernel. Both retain the same observations, actions, interventions, and metrics.

The main empirical lesson is that CompulsionBench exposes more than a watch-time leaderboard. In the active-threshold profile, deterministic diversity heuristics attain the highest watch time but also higher over-cap exposure, while a simple session cap sharply bounds over-cap severity at modest watch-time cost. A reference 120-minute profile shows that the same proxy channel can be inactive, motivating explicit threshold and proxy-activation reporting.

Our contributions are:

1. We introduce CompulsionBench, a reproducible benchmark for auditing and mitigating compulsion-risk proxies in sequential recommendation under explicit and auditable model assumptions.
2. We separate observable compulsion-risk proxies from simulator-internal diagnostics and align the constrained track with a decomposable observable cost, enabling comparison of engagement-maximizing, constrained, and post-hoc mitigation policies.
3. We provide two plug-compatible user-response backends, a fully transparent parametric core and an LLM-augmented extension fused with that core, under a single benchmark interface and official scorecard.
4. We provide a public-log calibration protocol with released audit artifacts, and show empirically why calibration fit, threshold provenance, and proxy activation must be reported together.

2 Related Work

Engagement, value, and reflective preference. A recurring critique in recommendation is that observed engagement is not itself a reliable measure of what users value, would endorse on reflection, or prefer under reduced temptation (Ekstrand & Willemsen, 2016; Milli et al., 2021; Kleinberg et al., 2024; Agarwal et al., 2024; Anwar et al., 2025). CompulsionBench turns that critique into

an evaluation problem: policies should be compared not only on engagement, but also on explicit long-horizon risk proxies.

Sequential recommendation and simulation. Work on preference dynamics and sequential recommendation shows that policies can change future behavior rather than merely predict immediate clicks (Kalimeris et al., 2021; Dean & Morgenstern, 2022; Curmei et al., 2022). RL-based recommender methods increasingly optimize long-horizon objectives such as engagement, return, and retention (Zou et al., 2019; Xue et al., 2023; Liu et al., 2024). Simulation platforms such as RecSim, RecSim NG, KuaiSim, and EasyRL4Rec make these sequential decision problems tractable under controlled conditions (Ie et al., 2019; Mladenov et al., 2021; Zhao et al., 2023; Yu et al., 2024), while KuaiRand and KuaiRec support public-log calibration targets (Gao et al., 2022b;a).

Safe RL, interventions, and psychology-informed dynamics. Safe and constrained RL provides the template for our observable-risk track, including optimization under explicit costs rather than pure engagement reward (Achiam et al., 2017; Chow et al., 2018). Work on incentive shaping and break-oriented interventions motivates comparing unconstrained policies against mitigation baselines (Kasirzadeh & Evans, 2023; Saig & Rosenfeld, 2023; Chee et al., 2024). Our latent dynamics are lightweight abstractions inspired by dual-process and habit-based models from neuroscience, computational psychiatry, and digital self-control research (Daw et al., 2011; Keramati & Gutkin, 2013; Lyngs et al., 2019). They support controlled component analysis and interpretation, not clinical modeling.

LLM-based user simulation. Recent LLM-agent work motivates hybrid simulators that preserve an interpretable internal model while allowing language to shape behavior (Zuo et al., 2026). We adopt that design principle at the benchmark level: CompulsionBench-LLM acts as an optional semantic robustness backend over the same sequential environment, rather than redefining the benchmark. This complements policy-facing calls for addiction-risk benchmarks and harm dashboards for highly personalized systems (Khraishi et al., 2025).

3 The CompulsionBench Benchmark

Two aligned backends. CompulsionBench is defined by a shared observation space, action space, intervention set, scorecard, and task family. The released environment therefore has two plug-compatible user-response backends: CompulsionBench-Param, the reference parametric simulator specified by closed-form equations, and CompulsionBench-LLM, an LLM-augmented extension that keeps the same state and action interface and changes only the user-response kernel. Formally, both backends implement

$$(r_t, y_t, s_{t+1}) \sim \mathcal{P}_B(\cdot \mid s_t, a_t), \qquad B \in \{\mathrm{Param}, \mathrm{LLM}\},$$

where $y_t \in \{0, 1\}$ is the continue/stop decision. When $y_t = \mathrm{stop}$, the stop branch additionally samples an off-platform gap $\Delta_t \sim P_{\mathrm{gap}}$ before forming s_{t+1}; no gap variable is sampled on the continue branch. The benchmark definition, tasks, and metrics are shared across B; only the response kernel differs.

3.1 Interface

Each episode represents a fixed interaction horizon, one day in the released profiles, during which a platform repeatedly recommends content to a user. The episode may contain multiple sessions separated by off-platform gaps. We model CompulsionBench as a partially observed sequential decision problem with hidden state

$$s_t = (\theta_i, h_t, c_t, f_t, \tau_t, \hat{r}_t, \hat{u}_t, \mathcal{H}_t),$$

where θ_i collects user-specific static parameters, h_t is a habit-like reinforcement trace, c_t is a control-capacity state, f_t is a consumption-load/fatigue-like state, τ_t is elapsed time, $\hat{r}_t$ is internal reward expectation, $\hat{u}_t$ is the platform's current preference estimate, and $\mathcal{H}_t$ is a short interaction history. The policy observes only

$$o_t = \left(\tau_t, \bar{\tau}_t, \hat{u}_t, \mathcal{H}_t^{\mathrm{obs}}, \{x_z\}_{z=1}^{Z}\right),$$

where $\bar{\tau}_t$ is wall-clock time and $\mathcal{H}_t^{\text{obs}}$ exposes recent-archetype summaries, recent watch-time summaries, current-session minutes ℓ_t, current-session item count j_t, most recent gap g_t, and wrapper flags. The latent internal variables $(h_t, c_t, f_t, \hat{r}_t)$ remain hidden. We retain the implementation symbols and use habit, control, and fatigue as shorthand for these synthetic states, not measured psychological attributes.

At each step, the platform chooses an action

$$a_t = (z_t, \lambda_t, \nu_t),$$

where $z_t \in \{1, \dots, Z\}$ is a content archetype, $\lambda_t \in \{0, \dots, P\}/P$ is a discretized personalization-intensity control, and $\nu_t \in \{0, 1\}$ is a variability mode that selects the variance of a mean-one multiplicative factor. The personalization control mixes the chosen archetype with the platform's belief:

$$\tilde{x}_t = (1 - \lambda_t) x_{z_t} + \lambda_t \frac{\hat{u}_t}{\|\hat{u}_t\|_2 + \epsilon}. \tag{1}$$

Higher λ_t therefore corresponds to stronger personalization toward the platform's current belief.

3.2 Response backends and interventions

The reference backend CompulsionBench-Param uses a deliberately lightweight user model in which watch time depends on content match, novelty, accumulated habit, fatigue, and stochastic reward variability. The design preserves partial observation, policy–response feedback, and recovery between sessions under a common, inspectable response model (Ie et al., 2019; Mladenov et al., 2021). Its particular functional forms are experimental assumptions; Appendix A provides the equations and relevant literature:

$$r_t = \min\Big\{ r_{\max},\ \text{scale} \cdot \text{softplus}\Big(\alpha \langle u_i, \tilde{x}_t \rangle + \beta n_i \,\text{Nov}_t(z_t) + \gamma_h h_t - \gamma_f f_t \Big) \cdot \eta_t(\nu_t) \cdot \xi_t \Big\}. \tag{2}$$

Here $\text{Nov}_t(z_t) = \exp(-\lambda_{\text{rep}} N_t(z_t))$, where $N_t(z_t)$ counts occurrences in the last k recommendations; $\eta_t(\nu_t)$ and ξ_t are mean-one log-normal factors. Write m_t^{base} for the expression inside softplus. Continuation uses a chosen mixture of reinforcement and deliberative/outside-option channels:

$$p_t(\text{continue}) = \sigma\Big(w_t^+ \tanh(Q_t^{\text{R},+}) + (1 - w_t^+) \tanh(Q_t^{\text{M},+}) - \text{friction}_t \Big). \tag{3}$$

Here $+$ denotes the post-consumption state; $Q_t^{\text{R},+}$ and $Q_t^{\text{M},+}$ are the reinforcement and deliberative/outside-option scores, and w_t^+ is their mixing weight. Define $q_t^{\text{base},+} = w_t^+ \tanh(Q_t^{\text{R},+}) + (1 - w_t^+) \tanh(Q_t^{\text{M},+})$, before the friction term and final sigmoid. Appendix A specifies these components.

Event order and re-entry. The simulator selects content, realizes watch time, updates the post-consumption state, and then draws continue/stop (subject to wrappers). On stopping, it records the session and samples $\Delta_t \sim P_{\text{gap}}$, the calibrated parametric log-normal gap law specified in Appendix A. The clock advances, latent states recover, session counters reset, and another recommendation begins only if the one-day horizon remains. Return metrics exclude the terminal censored gap.

CompulsionBench-LLM preserves the same latent state, observation, action, intervention wrappers, and scorecard. Following hybrid agent designs that combine an interpretable internal model with an external language layer (Zuo et al., 2026), it adds two semantic scores: a pre-consumption watch prior $a_t^{\text{watch}} \in [0, 1]$ and a post-consumption continuation prior $a_t^{\text{cont}} \in [0, 1]$, conditioned on a deterministic pseudo-persona $\mathcal{P}_i = \text{Render}(\theta_i)$, content card $\mathcal{D}_t$, and phase-specific state summary $\mathcal{C}_t$. The persona is a coarse textual rendering of sampled numerical parameters, not a fitted real-user profile. The priors are mapped to scores m_t^{LLM} and q_t^{LLM} on the PARAM scale and fused as follows:

$$\tilde{m}_t = (1 - \omega_{r,\text{llm}}) m_t^{\text{base}} + \omega_{r,\text{llm}} m_t^{\text{LLM}}, \tag{4}$$

$$\tilde{q}_t = (1 - \omega_{c,\text{llm}}) q_t^{\text{base},+} + \omega_{c,\text{llm}} q_t^{\text{LLM}}. \tag{5}$$

The stop-branch gap law, recovery equations, platform belief update, intervention wrappers, and evaluation metrics are unchanged. When $\omega_{r,\text{llm}} = \omega_{c,\text{llm}} = 0$, CompulsionBench-LLM reduces

exactly to CompulsionBench-Param. Substantive claims about backend sensitivity require the validity battery in Appendix B; the main results below do not rely on nonzero semantic-fusion effects.

The main text studies one post-hoc evaluation-time mitigation wrapper: a *session cap* applied to a learned checkpoint that forcibly terminates a session once cumulative session watch time exceeds T_{cap}. The realized watch time of the current item is not truncated. If an item pushes the session beyond T_{cap}, the session ends immediately after that item and may overshoot by at most one item. Appendix F describes optional wrappers such as break prompts (Saig & Rosenfeld, 2023), autoplay-off, and personalization throttles.

3.3 Scorecard and tasks

Let L denote session length in minutes, G the gap between consecutive sessions, $S_t = \ell_t$ cumulative watch time in the current session before step t, $\bar{\tau}_t$ wall-clock time at the start of step t, and r_t watch time accrued during the step. Official benchmark sessions are latent simulator sessions. The inactivity threshold Δ_{sess} is used only to extract empirical calibration targets from public logs and for calibration sensitivity analysis; it is not used to re-sessionize benchmark rollouts or leaderboard metrics.

The released scorecard focuses on observable outcomes: cumulative watch time, tail session risk, a profile-specific promoted night-use proxy when the orthogonality audit supports one, and over-cap exposure. Only observable quantities enter benchmark tasks and leaderboards; simulator-internal reflective-stop violations and related latent-state measures are appendix diagnostics. We report

$$\textsc{CumWatch} = \mathbb{E}\left[\sum_t r_t\right], \tag{6}$$

$$\text{CVaR}_{0.95}(L) = \mathbb{E}[L \mid L \geq \text{VaR}_{0.95}(L)], \tag{7}$$

$$\textsc{OverCapMinutes} = \mathbb{E}\left[\sum_t \{(S_t + r_t - T_{\text{ref}})_+ - (S_t - T_{\text{ref}})_+\}\right]. \tag{8}$$

The benchmark additionally computes NightMinutes, NightFraction, and LateNightSessionStartRate. A night proxy is promoted to the main text only if its policy-mean correlation with CumWatch is below the release threshold and it changes at least one policy comparison relative to plain watch time. Rapid-return metrics are retained as appendix sensitivity diagnostics.

Task 1: Engagement maximization. The first task optimizes cumulative watch time, $\max_\pi J_R(\pi) = \max_\pi \mathbb{E}_\pi[\sum_t r_t]$, and asks whether watch-time maximization changes compulsion-risk proxies. This follows the long-horizon recommender setting studied in engagement and retention optimization (Zou et al., 2019; Xue et al., 2023; Liu et al., 2024).

Task 2: Observable-risk constrained optimization. The second task optimizes reward subject to an observable per-step over-cap cost:

$$c_t^{\text{over}} = \big(S_t + r_t - T_{\text{ref}}\big)_+ - \big(S_t - T_{\text{ref}}\big)_+. \tag{9}$$

The optimization problem is

$$\max_\pi \; J_R(\pi) \quad \text{s.t.} \quad \mathbb{E}_\pi\left[\sum_t c_t^{\text{over}}\right] \leq \kappa_{\text{over}}.$$

This cost matches OverCapMinutes by construction, so the constrained track remains aligned with an observable leaderboard metric. We use it as a benchmark track inspired by constrained and safe RL formulations (Achiam et al., 2017; Chow et al., 2018).

4 Benchmark Setup

4.1 Calibration

The benchmark is synthetic, but we use public short-video data to define calibration targets and release explicit fit audits. We use KuaiRand as the primary source of sequential interaction data and

KuaiRec as a secondary source for preference heterogeneity and watch-time sanity checks (Gao et al., 2022b;a). Calibration is performed only at the level of aggregate statistics; we do not fit individual users.

Let $\mathcal{T}$ denote the empirical target statistics extracted from public logs. The default calibration targets are the session-length distribution in minutes, the stopping-hazard curve, the inter-session gap distribution, time-of-day session starts, and mean watch time across coarse content clusters. For these public-log targets only, session boundaries are extracted with an inactivity threshold Δ_{sess}, with sensitivity analysis over multiple thresholds. Let Θ_{cal} denote the subset of simulator parameters and user-prior hyperparameters selected for calibration. For a candidate parameter setting θ, let $\hat{\mathcal{T}}(\theta)$ be the corresponding simulated statistics. A simplified view of the fit objective is

$$\mathcal{L}_{\text{cal}}(\theta) = w_1 \,\text{KS}\Big(\hat{P}_L^\theta, P_L\Big) + w_2 \left\|\hat{h}_{\text{stop}}^\theta - h_{\text{stop}}\right\|_2^2 + w_3 \,\text{KS}\Big(\hat{P}_G^\theta, P_G\Big) + w_4 \sum_c \left(\hat{\mu}_r^\theta(c) - \mu_r(c)\right)^2. \tag{10}$$

The released operational objective is lexicographic rather than a flat weighted sum: it prioritizes session-length and session-item-count fit before gaps, normalized hazards, diurnal patterns, cluster watch means, and conditional-return summaries. Appendix C reports the full audit.

The calibration audit is reported as a benchmark artifact rather than hidden behind a single pass/fail label. The current simulator captures some aggregate structure but remains too light-tailed to serve as a close empirical twin of the public logs. We therefore use calibration to document the operating envelope of the benchmark, and we use prespecified fixed-paper threshold profiles for the main experiments rather than simulator-relative thresholds. Appendix C describes the calibration diagnostics.

4.2 Methods and evaluation

We report two fixed-paper evaluation profiles. The main active-threshold profile uses $T_{\text{ref}} = T_{\text{cap}} = 15$ and cap grid $\{10, 15, 20\}$. This is an operational stress-test threshold, not a clinical boundary or public-log quantile. T_{ref} measures exposure whereas T_{cap} controls intervention; equality is a default choice. The reference 120-minute profile uses $T_{\text{ref}} = T_{\text{cap}} = 120$ and cap grid $\{90, 120, 150\}$. This profile is used as a transparent sanity check on threshold activation.

The deterministic main-text scorecard compares RoundRobinPolicy, LeastRecentPolicy, NoveltyGreedyPolicy, a deterministic myopic ranker, PPO (Schulman et al., 2017), Lagrangian PPO, and a post-hoc session-cap wrapper applied at evaluation time to a learned PPO checkpoint. A companion stochastic scorecard evaluates Random, stochastic Myopic, stochastic PPO, stochastic Lagrangian PPO, and the same cap wrapper under stochastic rollout. All methods operate under partial observation and are denied access to latent state variables and user-type labels. PPO optimizes cumulative watch time; Lagrangian PPO optimizes the over-cap constrained objective and is interpreted as a constrained baseline rather than as a definitive safe-RL solution. Appendix D reports training budgets, checkpoint selection, and convergence diagnostics.

5 Results

The results use CompulsionBench as an evaluation object rather than as a leaderboard alone. We first establish the calibration boundary, then show the reward-risk scorecard, mitigation trade-offs, mechanism ablations, and a reference-threshold sanity check. Appendix G reports stochastic scorecards, cap-severity diagnostics, proxy audits, and training-convergence figures.

5.1 Calibration validity

The calibration artifact bundle compares public-log aggregate targets against the fitted simulator family for session lengths, stopping hazards, inter-session gaps, time-of-day session starts, and content-cluster watch means. The key result is not that the simulator is a close empirical twin, but that the release makes fit quality auditable. The current family is deliberately bounded and remains lighter-tailed than the public-log session tail, so calibration is used to set the interpretation boundary for the benchmark rather than to establish external prevalence or clinical validity.

Table 1: Deterministic active-threshold scorecard. Values are means ± episode-bootstrap 95% confidence intervals. Watch abbreviates CUMWATCH; CVaR is $\mathrm{CVaR}_{0.95}(L)$; OverCap abbreviates OVERCAPMINUTES; NightStart abbreviates LATENIGHTSESSIONSTARTRATE.

Policy	Watch ↑	CVaR ↓	OverCap ↓	NightStart ↓
Lagrangian PPO	60.56 ± 0.92	18.46 ± 0.34	2.20 ± 0.18	0.251 ± 0.005
LeastRecentPolicy	76.26 ± 1.07	21.59 ± 0.33	4.64 ± 0.27	0.252 ± 0.005
Myopic	65.09 ± 0.98	19.68 ± 0.29	3.00 ± 0.22	0.253 ± 0.004
NoveltyGreedyPolicy	76.51 ± 1.12	21.56 ± 0.33	4.61 ± 0.29	0.254 ± 0.004
PPO	63.26 ± 1.02	19.49 ± 0.30	2.84 ± 0.19	0.253 ± 0.004
PPO + SessionCap(15)	60.69 ± 0.96	15.71 ± 0.05	0.45 ± 0.04	0.253 ± 0.005
RoundRobinPolicy	77.82 ± 1.09	21.57 ± 0.31	4.63 ± 0.27	0.250 ± 0.004

This boundary strengthens the evaluation: all results below are within-benchmark comparisons under fixed scorecards and disclosed thresholds. Appendix C reports the calibration audit and its interpretation. In the main text, we rely only on the conclusion needed for benchmark interpretation: the released profiles are controlled evaluation settings, not empirical claims about real-world rates.

5.2 RISK UNDER ENGAGEMENT MAXIMIZATION

The deterministic scorecard in Table 1 is the main-text comparison for the active-threshold profile. The results do *not* show a monotone PPO-over-baseline ordering on cumulative watch time. The deterministic diversity heuristics exceed PPO on CUMWATCH: RoundRobinPolicy reaches 77.82 minutes, NoveltyGreedyPolicy 76.51, and LeastRecentPolicy 76.26, compared with 63.26 for PPO. These higher-watch policies also have higher over-cap exposure, around 4.6 minutes compared with 2.84 for PPO, and higher tail-session risk. The benchmark is therefore informative even when PPO is not the top watch-time policy, because the scorecard exposes tail-risk and mitigation diagnostics under matched evaluation rules.

The promoted night-use proxy, LATENIGHTSESSIONSTARTRATE, is close to 0.25 for all deterministic policies. We report it because the active-threshold proxy audit promotes it as less collinear with watch time than raw night minutes, but we do not interpret the small policy differences as a strong night-use effect.

Figure 1 plots the paired engagement and over-cap metrics, including the effect of applying a mitigation to the same learned policy.

5.3 MITIGATION TRADEOFFS

The active over-cap channel allows the benchmark to evaluate mitigation strategies. The constrained-RL baseline provides a partial deterministic improvement: Lagrangian PPO reduces OVERCAPMINUTES from 2.84 for PPO to 2.20, a reduction of approximately 22%, while reducing cumulative watch time from 63.26 to 60.56 minutes. This makes Lagrangian PPO a useful constrained baseline, but not a strong positive result: its operating point is Pareto-dominated by the hard session cap in the deterministic scorecard.

The post-hoc session cap is the strongest tested mitigation in the active-threshold run. PPO + SessionCap(15) reduces over-cap exposure from 2.84 to 0.45 minutes under deterministic evaluation, an 84% reduction, while reducing watch time by about 4.1%. It also reduces $\mathrm{CVaR}_{0.95}(L)$ from 19.49 to 15.71. The residual nonzero over-cap exposure is expected because the wrapper acts after the realized item watch time and therefore cannot truncate the final item that crosses the threshold.

A closer episode-level audit reveals why this mitigation is informative for benchmarking. The cap does not primarily prevent episodes from ever crossing the threshold: the deterministic positive-over-cap rate is 44.1% for both PPO and PPO + SessionCap(15), and the stochastic rate changes only from 55.2% to 55.5%. Instead, the cap bounds severity after crossing. Deterministically, episode-level over-cap p95 falls from 14.23 to 2.36 minutes and p99 from 24.90 to 4.27; stochastically, p95 falls from 18.55 to 2.11 and p99 from 28.83 to 3.65. Short-gap diagnostics do not show a compensating immediate re-entry spike: returns within one minute remain zero and returns within five minutes

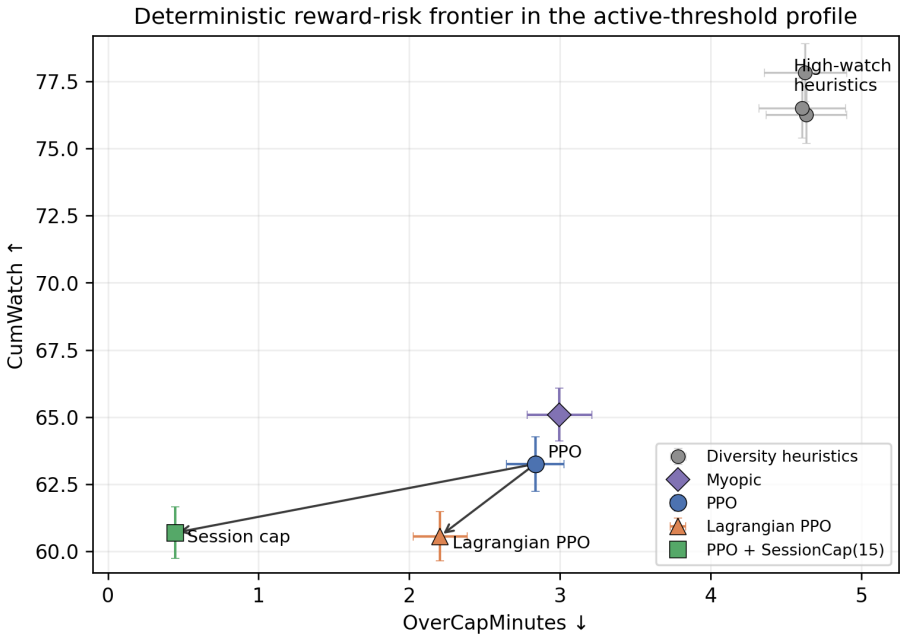

Figure 1: Deterministic reward-risk frontier for the active-threshold profile. The diversity heuristics occupy the high-watch, high-over-cap region. Lagrangian PPO moves PPO slightly left at some watch-time cost, while the post-hoc session cap moves PPO much further left with a modest watch-time cost.

Table 2: RL and mitigation operating points under deterministic and stochastic evaluation. Values are means; full confidence intervals are in Tables 1 and 11.

Policy	Det. Watch	Det. OverCap	Stoch. Watch	Stoch. OverCap
PPO	63.26	2.84	75.28	4.09
Lagrangian PPO	60.56	2.20	74.56	3.96
PPO + SessionCap(15)	60.69	0.45	71.96	0.50

remain near zero. Thus CompulsionBench distinguishes incidence prevention from tail-severity suppression, a useful property for mitigation audits.

The stochastic companion scorecard supports the same conclusion for the hard cap but not for constrained RL. Under stochastic evaluation, PPO obtains 75.28 minutes of watch time and 4.09 over-cap minutes. Lagrangian PPO obtains 74.56 minutes and 3.96 over-cap minutes, a small reduction relative to uncertainty. PPO + SessionCap(15) reduces over-cap exposure to 0.50 minutes at 71.96 minutes of watch time, corresponding to an 88% over-cap reduction at a 4.4% watch-time cost. We therefore treat Lagrangian PPO as a constrained baseline and sensitivity analysis rather than as evidence that standard constrained RL reliably recovers a robust feasible frontier.

5.4 Structural component analysis

Table 3 reports changes in OverCapMinutes for the NoHabit, NoVar, NoPers, and HomogeneousUsers conditions. Disabling reinforcement-trace accumulation, variability, or user heterogeneity reduces PPO over-cap exposure. For Lagrangian PPO, NoHabit and HomogeneousUsers reduce exposure, while NoVar has no effect at this operating point. These interventions assess the contributions of components within the specified simulator.

NoBeliefMixing (NoPers in the implementation) projects the fixed trained policy's λ_t coordinate to zero at evaluation; it does not retrain without personalization. Archetype selection z_t and variability ν_t remain available. The small effect therefore concerns belief mixing for these fixed policies, not

Table 3: Structural component ablations in the active-threshold profile. NoBeliefMixing is denoted NoPers in the implementation. Values report percentage changes in OVERCAPMINUTES relative to the corresponding default policy, with released 95% confidence intervals.

Ablation	PPO	Lagrangian PPO
NoHabit	$-24.75 \pm 4.05\%$	$-11.03 \pm 2.59\%$
NoVar	$-23.40 \pm 5.53\%$	$0.00 \pm 0.00\%$
NoBeliefMixing (NoPers)	$-0.83 \pm 1.95\%$	$-0.11 \pm 0.27\%$
HomogeneousUsers	$-21.07 \pm 2.90\%$	$-18.86 \pm 3.38\%$

personalization generally. It does not distinguish limited use of λ_t from weak response sensitivity, or establish what retraining would learn.

5.5 PARAM-120 SANITY CHECK

The reference 120-minute profile complements the active-threshold profile by checking proxy activation. With $T_{\text{ref}} = T_{\text{cap}} = 120$, every evaluated policy has OVERCAPMINUTES$= 0$, and the session-cap wrapper is inactive across the evaluated cap grid. This does not undermine the benchmark; it shows that a scorecard channel can be inactive under a particular threshold and simulator tail. Benchmark reports should therefore disclose threshold provenance and activation diagnostics rather than reporting zeros without context.

6 LIMITATIONS, ETHICS, AND RELEASE DESIGN

COMPULSIONBENCH is intended as a comparative benchmark rather than a new objective or a general-purpose simulator. It does not estimate real-world prevalence, establish causal claims about any deployed platform, or validate the simulator's latent states as real user psychology. Calibration is limited to aggregate public-log statistics, and simulator-internal diagnostics remain post-hoc tools for mechanism analysis rather than benchmark targets. COMPULSIONBENCH-LLM is a semantic perturbation of the synthetic response kernel. Its PARAM-derived personas and contexts support sensitivity analysis rather than independent behavioral validation.

The ethical release choice is to scope the benchmark around reproducible auditing rather than deployment guidance. The public reference package includes the default environment, released calibration manifests and audit tables, fixed evaluation scripts, deterministic seeding, result summaries, and documentation of assumptions, metrics, and failure modes. The appendix benchmark card and misuse statement document intended use, non-goals, reporting requirements, and foreseeable misuse modes.

7 CONCLUSION

COMPULSIONBENCH turns a difficult concern about engagement optimization into an executable evaluation problem: under explicit simulator assumptions, which policies increase observable compulsion-risk proxies, and which mitigations reduce them? The benchmark contributes a shared partially observed environment, two plug-compatible response backends, a scorecard that separates observable proxies from latent diagnostics, and two official tasks spanning engagement maximization and observable-risk constrained optimization.

The released results illustrate why this benchmark framing is useful. High-watch policies can occupy high-risk regions of the scorecard, simple caps can bound over-cap severity at modest engagement cost, constrained RL provides a useful but not yet dominant baseline, and threshold choices can render a proxy channel inactive. These are not clinical or deployment claims. They are reproducible within-benchmark audit findings designed to make future work on recommender-risk mitigation more comparable, falsifiable, and transparent.

AI USE STATEMENT

Generative AI was used to aid and polish writing, support research ideation, and draft sections of the manuscript. The optional LLM backend used in the research is described in Section 3.2 and Appendix B. The authors are responsible for the final text, claims, citations, and artifacts.

REPRODUCIBILITY STATEMENT

Code: https://github.com/anonymous-compulsionbench/CompulsionBench. Sections 3–4 and Appendices A–D describe the interface, assumptions, calibration scope, and evaluation protocol.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 22–31, 2017. URL https://proceedings.mlr.press/v70/achiam17a.html.

Arpit Agarwal, Nicolas Usunier, Alessandro Lazaric, and Maximilian Nickel. System-2 recommenders: Disentangling utility and engagement in recommendation systems via temporal point-processes. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, pp. 1763–1773, 2024. doi: 10.1145/3630106.3659004.

Md Sanzeed Anwar, Paramveer S. Dhillon, and Grant Schoenebeck. Recommendation and temptation. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, pp. 422–431, 2025. doi: 10.1145/3705328.3748063.

Mattia Bonzanni, Nicolas Rouleau, Michael Levin, and David Lee Kaplan. On the generalization of habituation: How discrete biological systems respond to repetitive stimuli. *BioEssays*, 41(7): 1900028, 2019. doi: 10.1002/bies.201900028.

Romain D. Cazé and Matthijs A. A. van der Meer. Adaptive properties of differential learning rates for positive and negative outcomes. *Biological Cybernetics*, 107(6):711–719, 2013. doi: 10.1007/s00422-013-0571-5.

Jerry Chee, Shankar Kalyanaraman, Sindhu Kiranmai Ernala, Udi Weinsberg, Sarah Dean, and Stratis Ioannidis. Harm mitigation in recommender systems under user preference dynamics. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 255–265, 2024. doi: 10.1145/3637528.3671925.

Yinlam Chow, Ofir Nachum, Edgar A. Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 8103–8112, 2018.

Mihaela Curmei, Andreas A. Haupt, Benjamin Recht, and Dylan Hadfield-Menell. Towards psychologically-grounded dynamic preference models. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pp. 35–48, 2022. doi: 10.1145/3523227.3546778.

Nathaniel D. Daw, Samuel J. Gershman, Ben Seymour, Peter Dayan, and Raymond J. Dolan. Model-based influences on humans' choices and striatal prediction errors. *Neuron*, 69(6):1204–1215, 2011. doi: 10.1016/j.neuron.2011.02.027.

Sarah Dean and Jamie Morgenstern. Preference dynamics under personalized recommendations. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pp. 795–816, 2022. doi: 10.1145/3490486.3538346.

Michael D. Ekstrand and Martijn C. Willemsen. Behaviorism is not enough: Better recommendations through listening to users. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 221–224, 2016. doi: 10.1145/2959100.2959179.

Chongming Gao, Shijun Li, Wenqiang Lei, Jiawei Chen, Biao Li, Peng Jiang, Xiangnan He, Jiaxin Mao, and Tat-Seng Chua. Kuairec: A fully-observed dataset and insights for evaluating recommender systems. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, pp. 540–550, 2022a. doi: 10.1145/3511808.3557220.

Chongming Gao, Shijun Li, Yuan Zhang, Jiawei Chen, Biao Li, Wenqiang Lei, Peng Jiang, and Xiangnan He. Kuairand: An unbiased sequential recommendation dataset with randomly exposed videos. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, pp. 3953–3957, 2022b. doi: 10.1145/3511808.3557624.

Eugene Ie, Chih-Wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. Recsim: A configurable simulation platform for recommender systems. *arXiv preprint arXiv:1909.04847*, 2019. URL https://arxiv.org/abs/1909.04847.

Michael Inzlicht, Brandon J. Schmeichel, and C. Neil Macrae. Why self-control seems (but may not be) limited. *Trends in Cognitive Sciences*, 18(3):127–133, 2014. doi: 10.1016/j.tics.2013.12.009.

Dimitris Kalimeris, Smriti Bhagat, Shankar Kalyanaraman, and Udi Weinsberg. Preference amplification in recommender systems. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 805–815, 2021. doi: 10.1145/3447548.3467298.

Atoosa Kasirzadeh and Charles Evans. User tampering in reinforcement learning recommender systems. In *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 58–69, 2023. doi: 10.1145/3600211.3604669.

Mehdi Keramati and Boris Gutkin. Imbalanced decision hierarchy in addicts emerging from drug-hijacked dopamine spiraling circuit. *PLOS ONE*, 8(4):e61489, 2013. doi: 10.1371/journal.pone.0061489.

Raad Khraishi, Cristovao Iglesias, Devesh Batra, Peter Gostev, Giulio Pelosio, Ramin Okhrati, and Greig A. Cowan. Real-time hyper-personalized generative ai should be regulated to prevent the rise of "digital heroin". In *NeurIPS 2025 Position Paper Track*, 2025.

Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. The challenge of understanding what users want: Inconsistent preferences and engagement optimization. *Management Science*, 70 (9):6336–6355, 2024. doi: 10.1287/mnsc.2022.03683.

Sang Wan Lee, Shinsuke Shimojo, and John P. O'Doherty. Neural computations underlying arbitration between model-based and model-free learning. *Neuron*, 81(3):687–699, 2014. doi: 10.1016/j.neuron.2013.11.028.

Ziru Liu, Shuchang Liu, Bin Yang, Zhenghai Xue, Qingpeng Cai, Xiangyu Zhao, Zijian Zhang, Lantao Hu, Han Li, and Peng Jiang. Modeling user retention through generative flow networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5497–5508, 2024. doi: 10.1145/3637528.3671531.

Ulrik Lyngs, Kai Lukoff, Petr Slovak, Reuben Binns, Adam Slack, Michael Inzlicht, Max Van Kleek, and Nigel Shadbolt. Self-control in cyberspace: Applying dual systems theory to a review of digital self-control tools. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–18, 2019. doi: 10.1145/3290605.3300361.

Smitha Milli, Luca Belli, and Moritz Hardt. From optimizing engagement to measuring value. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 714–722, 2021. doi: 10.1145/3442188.3445933.

Martin Mladenov, Chih-Wei Hsu, Vihan Jain, Eugene Ie, Christopher Colby, Nicolas Mayoraz, Hubert Pham, Dustin Tran, Ivan Vendrov, and Craig Boutilier. Recsim ng: Toward principled uncertainty modeling for recommender ecosystems. *arXiv preprint arXiv:2103.08057*, 2021. URL https://arxiv.org/abs/2103.08057.

Tanja Müller, Miriam C. Klein-Flügge, Sanjay G. Manohar, Masud Husain, and Matthew A. J. Apps. Neural and computational mechanisms of momentary fatigue and persistence in effort-based choice. *Nature Communications*, 12:4593, 2021. doi: 10.1038/s41467-021-24927-7.

Eden Saig and Nir Rosenfeld. Learning to suggest breaks: Sustainable optimization of long-term user engagement. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 29671–29696, 2023. URL https://proceedings.mlr.press/v202/saig23a.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Bruno Sguerra, Viet-Anh Tran, and Romain Hennequin. Discovery dynamics: Leveraging repeated exposure for user and music characterization. In *Proceedings of the 16th ACM Conference on Recommender Systems*, 2022. doi: 10.1145/3523227.3551474. URL https://arxiv.org/abs/2210.16226.

Wanqi Xue, Qingpeng Cai, Ruohan Zhan, Dong Zheng, Peng Jiang, Kun Gai, and Bo An. Resact: Reinforcing long-term engagement in sequential recommendation with residual actor. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=HmPOzJQhbwg.

Yuanqing Yu, Chongming Gao, Jiawei Chen, Heng Tang, Yuefeng Sun, Qian Chen, Weizhi Ma, and Min Zhang. Easyrl4rec: An easy-to-use library for reinforcement learning based recommender systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 977–987, 2024. doi: 10.1145/3626772.3657868.

Kesen Zhao, Shuchang Liu, Qingpeng Cai, Xiangyu Zhao, Ziru Liu, Dong Zheng, Peng Jiang, and Kun Gai. Kuaisim: A comprehensive simulator for recommender systems. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44880–44897, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c7f8f98f9a8f5650922dd4545254f28-Abstract-Datasets_and_Benchmarks.html.

Lixin Zou, Long Xia, Zhuoye Ding, Jiaxing Song, Weidong Liu, and Dawei Yin. Reinforcement learning to optimize long-term user engagement in recommender systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2810–2818, 2019. doi: 10.1145/3292500.3330668.

Fei Zuo, Kezhi Wang, Xiaomin Chen, and Yizhou Huang. Biollmagent: A hybrid framework with enhanced structural interpretability for simulating human decision-making in computational psychiatry. *arXiv preprint arXiv:2603.05016*, 2026.

A PARAMETRIC BACKEND EQUATIONS

This appendix collects the full transition equations, user priors, and shared default configuration for the COMPULSIONBENCH-PARAM backend. The latent state at step t is

$$s_t = (\theta_i, h_t, c_t, f_t, \tau_t, \hat{r}_t, \hat{u}_t, \mathcal{H}_t),$$

where θ_i contains user-specific static parameters, h_t is a habit-like reinforcement trace, c_t is a control-capacity state, f_t is a consumption-load/fatigue-like state, τ_t is elapsed time, $\hat{r}_t$ is internal reward expectation, $\hat{u}_t$ is the platform's current preference estimate, and $\mathcal{H}_t$ is a short interaction history. Each episode samples a user type

$$\theta_i = (u_i, v_i, c_{0,i}, n_i, \rho_{h,i}, \rho_{c,i}, \gamma_{rh,i}, \gamma_{rc,i}),$$

where $u_i \in \mathbb{R}^d$ is a preference embedding, $v_i \in [0,1]$ is a susceptibility scalar, $c_{0,i} \in [0,1]$ is baseline control capacity, $n_i \geq 0$ is novelty sensitivity, $\rho_{h,i}$ controls reinforcement-trace accumulation, $\rho_{c,i}$ controls control-capacity depletion, and $\gamma_{rh,i}, \gamma_{rc,i}$ are recovery rates during off-platform gaps. Table 4 summarizes the default parameter priors and configuration.

To recover wall-clock time from elapsed episode time, each episode also samples

$$\tau_{i,\text{start}} \sim \text{Uniform}(0, 1440), \qquad \bar{\tau}_t = (\tau_t + \tau_{i,\text{start}}) \bmod 1440.$$

All time-of-day quantities in the benchmark, including the outside option, night-use metrics, and observable-risk costs, are functions of $\bar{\tau}_t$ rather than τ_t. We use $\sigma(x) = (1+e^{-x})^{-1}$, $\text{softplus}(x) = \log(1+e^x)$, $\text{ReLU}(x) = \max\{x, 0\}$, $\text{clip}(x, \ell, u) = \min\{u, \max\{\ell, x\}\}$, and $\mathbb{I}[\cdot]$ for the indicator function.

A.1 SAMPLING AND INITIALIZATION

The reference implementation fixes the content catalog once per benchmark instance. For each archetype $z \in \{1, \ldots, Z\}$, draw

$$g_z \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_d), \qquad x_z = \frac{g_z}{\|g_z\|_2 + \epsilon},$$

with $\epsilon = 10^{-8}$. The catalog $\{x_z\}_{z=1}^{Z}$ is sampled once using the catalog seed in Table 4 and then reused across all episode seeds, training seeds, and evaluation runs for that benchmark instance; it is not resampled at every episode reset.

At episode reset, the simulator samples a user embedding by first drawing $g_i \sim \mathcal{N}(0, I_d)$, normalizing it, then sampling an anchor index $A_i \sim \text{Uniform}\{1, \ldots, Z\}$ and setting

$$u_i = \frac{g_i/(\|g_i\|_2 + \epsilon) + \sigma_u x_{A_i}}{\|g_i/(\|g_i\|_2 + \epsilon) + \sigma_u x_{A_i}\|_2 + \epsilon},$$

with $\sigma_u = 0.15$ in the default configuration. Thus u_i is normalized after sampling, and the anchor term adds mild catalog-aligned heterogeneity. The remaining user parameters are sampled independently from the priors in Table 4, with

$$\gamma_{rh,i} = \bar{\gamma}_{rh}\zeta_{rh,i}, \qquad \gamma_{rc,i} = \bar{\gamma}_{rc}\zeta_{rc,i}, \qquad \zeta_{rh,i}, \zeta_{rc,i} \sim \text{LogNormal}(0, 0.15^2).$$

Initial conditions at reset are

$$h_0 = 0, \quad c_0 = c_{0,i}, \quad f_0 = 0, \quad \hat{r}_0 = 0, \quad \hat{u}_0 = 0 \in \mathbb{R}^d, \quad \tau_0 = 0.$$

The finite-memory state is initialized as

$$\mathcal{H}_0 = (\mathcal{Z}_0, \mathcal{R}_0, \ell_0, j_0, g_0, \text{br}_0) = (\varnothing, \varnothing, 0, 0, 0, 0),$$

where $\mathcal{Z}_t$ stores the last k recommended archetype indices, $\mathcal{R}_t$ stores the corresponding last k realized watch times, ℓ_t is cumulative watch time in the current session, j_t is the current-session item count, g_t is the most recent off-platform gap, and $\text{br}_t \in \{0, 1\}$ indicates whether the break-prompt threshold is currently active. The buffers $\mathcal{Z}_t$ and $\mathcal{R}_t$ are truncated to their most recent k entries; when they contain fewer than k interactions, derived observation features use the available entries and zero-fill the missing positions. Concretely, the observation exposes the mean catalog embedding over $\mathcal{Z}_t$, the

normalized histogram of archetypes in $\mathcal{Z}_t$, the mean of $\mathcal{R}_t$, the last realized watch time, ℓ_t, j_t, g_t, and br_t, with normalization constants 10 minutes, 10 minutes, 120 minutes, 30 items, and 180 minutes respectively.

Calibration does not replace the gap process with an empirical sampler. In the released experiments, calibration fixes only the parameters of a parametric log-normal gap law, and the implemented simulator samples

$$\Delta_t \sim \mathrm{LogNormal}(\mu_g, \sigma_g^2)$$

with $(\mu_g, \sigma_g) = (4.2, 0.8)$ in the default calibrated instance. The reference benchmark code does not apply additional wall-clock modulation to P_{gap}.

Novelty and watch-time decomposition. Let $N_t(z) = \sum_{z' \in \mathcal{Z}_t} \mathbb{I}[z' = z]$ be the number of times archetype z appears in the last k recommended items. We define a recency-based novelty term as

$$\mathrm{Nov}_t(z) = \exp(-\lambda_{\mathrm{rep}} N_t(z)), \tag{11}$$

so that recently repeated content is less novel. Exponential habituation motivates this modeling family (Bonzanni et al., 2019), while inverted-U repeated-exposure effects provide an alternative pattern (Sguerra et al., 2022). Given action a_t, the corresponding watch-time logit is

$$m_t = \alpha \langle u_i, \tilde{x}_t \rangle + \beta n_i \mathrm{Nov}_t(z_t) + \gamma_h h_t - \gamma_f f_t. \tag{12}$$

Define

$$\eta_t(\nu_t) \sim \mathrm{LogNormal}\big(-\tfrac{1}{2}\sigma^2_{\eta,\nu_t}, \sigma^2_{\eta,\nu_t}\big), \qquad \sigma_{\eta,\nu_t} = \begin{cases} \sigma_{\eta,0}, & \nu_t = 0, \\ \sigma_{\eta,1}, & \nu_t = 1, \end{cases}$$

and

$$\xi_t \sim \mathrm{LogNormal}\big(-\tfrac{1}{2}\sigma^2_{\xi}, \sigma^2_{\xi}\big).$$

Thus ν_t changes watch-time dispersion through a mean-one multiplicative factor rather than by shifting the watch-time logit. Watch time in minutes is then generated by Eq. (2).

Post-consumption intermediate state. Asymmetric prediction-error learning (Cazé & van der Meer, 2013), motivational accounts of self-control (Inzlicht et al., 2014), and effort-related fatigue (Müller et al., 2021) provide conceptual context for these reduced-form states, without specifying their exact update rules. After realizing watch time r_t, the simulator forms the post-consumption intermediate variables

$$\hat{r}_t^+ = (1 - \lambda_r)\hat{r}_t + \lambda_r r_t, \tag{13}$$

$$\delta_t = r_t - \hat{r}_t, \tag{14}$$

$$h_t^+ = \mathrm{clip}((1 - \rho_{h,i})h_t + \rho_{h,i}(1 + v_i)\mathrm{ReLU}(\delta_t), 0, h_{\max}), \tag{15}$$

$$c_t^+ = \max\Big\{0, c_t - \rho_{c,i}\chi_c \frac{r_t}{\mathrm{scale}}\Big\}, \tag{16}$$

$$f_t^+ = \min\Big\{1, (1 - \rho_f)f_t + \rho_f a_f \frac{r_t}{\mathrm{scale}}\Big\}, \tag{17}$$

$$\tau_t^+ = \tau_t + r_t, \tag{18}$$

$$\ell_t^+ = \ell_t + r_t, \tag{19}$$

$$j_t^+ = j_t + 1. \tag{20}$$

The reward prediction error is computed relative to the pre-consumption expectation $\hat{r}_t$, while the continue/stop decision is taken from the post-consumption state $(h_t^+, c_t^+, f_t^+, \tau_t^+, \hat{r}_t^+)$.

Reinforcement and deliberative/outside-option continuation components. Let $\bar{\tau}_t^+ = (\tau_t^+ + \tau_{i,\mathrm{start}}) \bmod 1440$. The reinforcement-channel score evaluated on the post-consumption state is

$$Q_t^{\mathrm{R},+} = \omega_h \frac{h_t^+}{h_{\max}} + \omega_r \frac{\hat{r}_t^+}{\mathrm{scale}}, \tag{21}$$

while the deliberative/outside-option score is

$$Q_t^{\mathrm{M},+} = \omega_{r,m} \frac{\hat{r}_t^+}{\mathrm{scale}} + \omega_c c_t^+ - \omega_f f_t^+ - \omega_o V^{\mathrm{out}}(\bar{\tau}_t^+). \tag{22}$$

The outside option varies with wall-clock time:

$$V^{\text{out}}(\bar{\tau}) = \bar{V} + A\cos\left(\frac{2\pi(\bar{\tau}-\phi)}{1440}\right). \tag{23}$$

The state-dependent mixing weight is

$$w_t^+ = \sigma\left(\kappa_0 + \kappa_h \frac{h_t^+}{h_{\max}} - \kappa_c c_t^+\right). \tag{24}$$

Together, Eqs. (21)–(24) define the reduced continue model in Eq. (3) with $Q_t^{\mathrm{R},+}$, $Q_t^{\mathrm{M},+}$, and w_t^+:

$$p_t(\text{continue}) = \sigma\Big(w_t^+ \tanh(Q_t^{\mathrm{R},+}) + (1-w_t^+)\tanh(Q_t^{\mathrm{M},+}) - \text{friction}_t\Big).$$

Controller arbitration (Lee et al., 2014) motivates combining distinct channels; the additive mixture here is a reduced-form modeling choice.

In the base environment, $\text{friction}_t = 0$ and $\text{br}_t = \text{br}_t^+ = 0$ for all t. Optional wrappers add nonnegative offsets to this base value. Under the default break-prompt wrapper with thresholds $(T_{\text{br}}, J_{\text{br}}) = (30\text{ min}, 20\text{ items})$, we replace the post-consumption activation flag by

$$\text{br}_t^+ = \mathbb{I}[\ell_t^+ \geq T_{\text{br}} \vee j_t^+ \geq J_{\text{br}}],$$

and add $0.5\,\text{br}_t^+$ to friction_t. The session-cap wrapper never truncates the current item: if it is active and $\ell_t^+ > T_{\text{cap}}$, the realized watch time r_t is kept and the wrapper overrides the sampled branch by forcing $y_t = \text{stop}$ after that item.

State updates and session boundaries. The continue/stop draw is taken after the post-consumption update above. The history buffers always append the realized interaction before branching:

$$\mathcal{Z}_{t+1}^{\text{hist}} = \text{Append}_k(\mathcal{Z}_t, z_t), \qquad \mathcal{R}_{t+1}^{\text{hist}} = \text{Append}_k(\mathcal{R}_t, r_t),$$

where Append_k appends the new entry and retains the most recent k elements. The platform belief update is branch-invariant:

$$\hat{u}_{t+1} = \text{ClipRad}_5\Big((1-\eta_u)\hat{u}_t + \eta_u \frac{r_t}{\text{scale}}\tilde{x}_t\Big), \tag{25}$$

where $\text{ClipRad}_5(v) = v$ if $\|v\|_2 \leq 5$ and $\text{ClipRad}_5(v) = 5v/\|v\|_2$ otherwise. If the user continues, no gap is sampled and the next state inherits the post-consumption variables:

$$h_{t+1} = h_t^+, \tag{26}$$
$$c_{t+1} = c_t^+, \tag{27}$$
$$f_{t+1} = f_t^+, \tag{28}$$
$$\tau_{t+1} = \tau_t^+, \tag{29}$$
$$\hat{r}_{t+1} = \hat{r}_t^+, \tag{30}$$
$$\ell_{t+1} = \ell_t^+, \tag{31}$$
$$j_{t+1} = j_t^+, \tag{32}$$
$$g_{t+1} = g_t, \tag{33}$$
$$\text{br}_{t+1} = \text{br}_t^+, \tag{34}$$
$$\mathcal{Z}_{t+1} = \mathcal{Z}_{t+1}^{\text{hist}}, \tag{35}$$
$$\mathcal{R}_{t+1} = \mathcal{R}_{t+1}^{\text{hist}}. \tag{36}$$

If the user stops, the current session ends and an off-platform gap Δ_t is sampled from the calibrated gap distribution P_{gap}. During the gap, latent variables recover from the post-consumption state:

$$h_{t+1} = h_t^+ e^{-\gamma_{rh,i}\Delta_t}, \tag{37}$$
$$c_{t+1} = c_t^+ + (c_{0,i} - c_t^+)\big(1 - e^{-\gamma_{rc,i}\Delta_t}\big), \tag{38}$$
$$f_{t+1} = f_t^+ e^{-\gamma_{rf}\Delta_t}, \tag{39}$$
$$\tau_{t+1} = \tau_t^+ + \Delta_t, \tag{40}$$
$$\hat{r}_{t+1} = \hat{r}_t^+, \tag{41}$$
$$\ell_{t+1} = 0, \tag{42}$$
$$j_{t+1} = 0, \tag{43}$$
$$g_{t+1} = \Delta_t, \tag{44}$$
$$\text{br}_{t+1} = 0, \tag{45}$$
$$\mathcal{Z}_{t+1} = \mathcal{Z}_{t+1}^{\text{hist}}, \tag{46}$$
$$\mathcal{R}_{t+1} = \mathcal{R}_{t+1}^{\text{hist}}. \tag{47}$$

Thus $\hat{r}_t$ does not decay during off-platform gaps: on both branches the next-step reward expectation is $\hat{r}_{t+1} = \hat{r}_t^+$. The completed stop-branch session records (ℓ_t^+, j_t^+) before the within-session counters are reset to zero for the next session. A session is defined as a maximal sequence of consecutive continue decisions, and the episode ends when τ_t exceeds the horizon H.

Episode rollout. The simulator executes each episode in the following order:

1. Sample or reuse the fixed catalog $\{x_z\}_{z=1}^{Z}$, sample user parameters θ_i and wall-clock offset $\tau_{i,\text{start}}$, and initialize $(h_0, c_0, f_0, \tau_0, \hat{r}_0, \hat{u}_0, \mathcal{H}_0)$ as in Appendix A.1.

2. While $\tau_t < H$, form the observation from $\bar{\tau}_t$, $\hat{u}_t$, and the derived features of $\mathcal{H}_t$, then choose an action $a_t = (z_t, \lambda_t, \nu_t)$.

3. If the backend is CompulsionBench-LLM, render the deterministic pseudo-persona $\mathcal{P}_i$ and archetype card $\mathcal{D}_t = \text{Render}(z_t, \lambda_t, \nu_t, \mathcal{H}_t, \bar{\tau}_t)$; otherwise skip this rendering step. Compute the watch score, realize watch time r_t, and form the post-consumption variables $(\hat{r}_t^+, h_t^+, c_t^+, f_t^+, \tau_t^+, \ell_t^+, j_t^+)$.

4. Compute $\bar{\tau}_t^+$, evaluate the continuation score using $(Q_t^{\text{R},+}, Q_t^{\text{M},+}, w_t^+)$ in CompulsionBench-Param or the fused continuation score in CompulsionBench-LLM, add any wrapper friction, and sample the continue/stop decision.

5. If the session-cap wrapper is active and $\ell_t^+ > T_{\text{cap}}$, override the sampled branch by forcing stop. If the resulting branch is stop, sample $\Delta_t \sim P_{\text{gap}}$ and apply the recovery equations; otherwise inherit the post-consumption state directly.

6. Update the history buffers $(\mathcal{Z}_{t+1}, \mathcal{R}_{t+1})$, the platform belief $\hat{u}_{t+1}$, the latest-gap field g_{t+1}, and the break flag br_{t+1} according to the branch-specific rules above; then emit reward r_t, observable costs, and the next observation o_{t+1}.

7. End the episode when $\tau_t \geq H$, finalize any horizon-censored terminal session, and aggregate the recorded sessions into the reported metrics.

Algorithm 1 One-step transition for CompulsionBench-Param / CompulsionBench-LLM

Require: Backend $B \in \{\mathrm{Param}, \mathrm{LLM}\}$, current state s_t, policy π, active wrappers $\mathcal{W}$

1: Form observation o_t from $(\bar{\tau}_t, \hat{u}_t, \mathcal{H}_t, \{x_z\}_{z=1}^{Z})$
2: Choose action $a_t = (z_t, \lambda_t, \nu_t) \sim \pi(\cdot \mid o_t)$
3: **if** $B = \mathrm{LLM}$ **then**
4: Render the deterministic pseudo-persona $\mathcal{P}_i = \mathrm{Persona}(\theta_i)$ and archetype card $\mathcal{D}_t = \mathrm{Render}(z_t, \lambda_t, \nu_t, \mathcal{H}_t, \bar{\tau}_t)$
5: **end if**
6: Compute the watch score using Eq. (12) for CompulsionBench-Param or Eqs. (48) and (4) for CompulsionBench-LLM
7: Realize watch time r_t from Eq. (2)
8: Update the post-consumption latent state $(\hat{r}_t^+, h_t^+, c_t^+, f_t^+, \tau_t^+, \ell_t^+, j_t^+)$ and set $\mathrm{br}_t^+ = 0$ in the base environment, replacing it by $\mathbb{I}[\ell_t^+ \geq T_{\mathrm{br}} \vee j_t^+ \geq J_{\mathrm{br}}]$ when the break wrapper is active
9: Compute the continuation score using Eq. (3) for CompulsionBench-Param or Eq. (5) for CompulsionBench-LLM, with $\mathrm{friction}_t = 0$ in the base environment and wrapper offsets added on top
10: Sample $y_t \in \{\mathrm{continue}, \mathrm{stop}\}$ from $p_t(\mathrm{continue})$
11: **if** $T_{\mathrm{cap}} \in \mathcal{W}$ and $\ell_t^+ > T_{\mathrm{cap}}$ **then**
12: Override the sampled branch by setting $y_t \leftarrow \mathrm{stop}$
13: **end if**
14: **if** $y_t = \mathrm{stop}$ **then**
15: Sample $\Delta_t \sim P_{\mathrm{gap}}$ and apply the stop-branch recovery equations for $(h_{t+1}, c_{t+1}, f_{t+1}, \tau_{t+1})$
16: Set $\hat{r}_{t+1} = \hat{r}_t^+$, $\ell_{t+1} = j_{t+1} = 0$, $g_{t+1} = \Delta_t$, and $\mathrm{br}_{t+1} = 0$
17: **else**
18: Set $(h_{t+1}, c_{t+1}, f_{t+1}, \tau_{t+1}, \hat{r}_{t+1}, \ell_{t+1}, j_{t+1}, g_{t+1}, \mathrm{br}_{t+1})$ $= (h_t^+, c_t^+, f_t^+, \tau_t^+, \hat{r}_t^+, \ell_t^+, j_t^+, g_t, \mathrm{br}_t^+)$
19: **end if**
20: Update $(\mathcal{Z}_{t+1}, \mathcal{R}_{t+1})$ by appending (z_t, r_t) and update the platform belief $\hat{u}_{t+1}$ with Eq. (25)
21: Emit reward r_t, observable costs $(c_t^{\mathrm{night}}, c_t^{\mathrm{over}})$, and the next observation o_{t+1}

Table 4: Default shared configuration for the benchmark topology and parametric core used by COMPULSIONBENCH-PARAM; COMPULSIONBENCH-LLM inherits these settings and adds the backend-specific defaults in Table 7. Parameters are grouped by role to make the reference instance easier to audit and reproduce.

Component	Default	Notes
Environment and action space		
Episode horizon H	1440 min	One-day episode; multiple sessions may occur within an episode.
Wall-clock offset $\tau_{i,\text{start}}$	$\text{Uniform}(0, 1440)$	Randomizes episode start time on the 24-hour clock.
Catalog seed	12345	Samples $\{x_z\}_{z=1}^{Z}$ once per benchmark instance and keeps it fixed across train/eval seeds.
Embedding dimension d	16	Latent user/content feature dimension.
Archetype count Z	30	Keeps the catalog expressive but the action space manageable.
History length k	10	Used for novelty and recent-history features.
Personalization bins P	5	$\lambda_t \in \{0, 0.2, \ldots, 1.0\}$.
Reward-variability modes	2	Low/high variability setting for ν_t.
Watch-time scale	1.0 min	Converts normalized engagement into minutes.
Max item watch time $r_{\max}$	5.0 min	Short-video-inspired per-item cap that prevents a single item from dominating session-tail metrics.
Numerical stability constant ϵ	10^{-8}	Used in vector normalizations and the personalization mixture.
Behavioral dynamics		
Variability scales $(\sigma_{\eta,0}, \sigma_{\eta,1})$	$(0.2, 0.8)$	Low/high dispersion of the mean-one multiplicative factor $\eta_t(\nu_t)$ in Eq. (2).
Residual multiplicative noise σ_ξ	0.2	Mean-one residual watch-time variability in Eq. (2).
Alignment / novelty (α, β)	$(1.0, 0.3)$	Default weights on content match and novelty.
Reinforcement trace / load (γ_h, γ_f)	$(0.5, 0.4)$	Carry-over weights in the watch-time logit.
Repeat-decay λ_{rep}	0.7	Controls novelty loss from recent repetition.
Reward smoothing λ_r	0.1	Update rate for internal reward expectation.
Trace cap $h_{\max}$	10.0	Prevents unbounded reinforcement-trace growth.
Platform update η_u	0.1	EMA rate for the platform preference estimate.
Belief radius clip	5.0	After Eq. (25), radially clip $\hat{u}_t$ if its norm exceeds 5.
Reinforcement weights (ω_h, ω_r)	$(0.6, 0.2)$	Trace and reward contribution to the reinforcement-channel score.
Deliberative weights $(\omega_{r,m}, \omega_c, \omega_f, \omega_o)$	$(0.4, 1.0, 0.6, 0.8)$	Reward, control capacity, consumption load, and outside-option terms.
Mixing logits $(\kappa_0, \kappa_h, \kappa_c)$	$(-2.0, 1.0, 1.0)$	Governs the shift from the deliberative to the reinforcement channel.
Outside option $(\bar{V}, A, \phi)$	$(0.8, 0.4, 840)$	Mean, amplitude, and phase; $\phi = 840$ corresponds to 14:00.

Continued on next page

Table 4 (continued)

Component	Default	Notes
Calibrated gap law P_{gap}	$\text{LogNormal}(4.2, 0.8^2)$	Exact implementation used in experiments; calibration tunes only these parameters and does not replace the law with an empirical or time-modulated sampler.
Load smoothing ρ_f	0.08	Governs the AR(1)-style load update during continued consumption.
Control depletion multiplier χ_c	0.25	Converts watched minutes into control-capacity depletion.
Load accumulation multiplier a_f	0.6	Converts watched minutes into consumption-load accrual.
User priors and recovery		
User embedding anchor weight σ_u	0.15	Weight on the sampled catalog anchor in the user-embedding initializer.
User prior v_i	$\text{Beta}(2, 5)$	Mostly low-to-moderate susceptibility.
User prior $c_{0,i}$	$\text{Beta}(8, 2)$	Typically high baseline control capacity.
User prior n_i	$\text{LogNormal}(-0.2, 0.25^2)$	Moderate novelty-seeking.
User prior $\rho_{h,i}$	$\text{LogNormal}(-3.0, 0.35^2)$	Heterogeneous reinforcement-trace accumulation rates.
User prior $\rho_{c,i}$	$\text{LogNormal}(-2.5, 0.35^2)$	Heterogeneous control-capacity depletion rates.
Recovery rates $(\bar{\gamma}_{rh}, \bar{\gamma}_{rc}, \gamma_{rf})$	$(0.005, 0.010, 0.008)$	$\gamma_{rh,i} = \bar{\gamma}_{rh}\zeta_{rh,i}$ and $\gamma_{rc,i} = \bar{\gamma}_{rc}\zeta_{rc,i}$ with $\zeta_{rh,i}, \zeta_{rc,i} \sim \text{LogNormal}(0, 0.15^2)$; γ_{rf} is global.
Initialization defaults	$h_0 = f_0 = \hat{r}_0 = 0$, $\tau_0 = 0$, $\hat{u}_0 = 0$, $c_0 = c_{0,i}$	History buffers start empty; see Appendix A.1.
Metrics and wrapper parameter defaults (inactive unless enabled)		
Night window $\mathcal{W}_{\text{night}}$	00:00–06:00	Used for night-use metrics and observable-risk costs.
Tail confidence level α_{tail}	0.95	Default level for $\text{CVaR}_{0.95}(L)$.
Tail quantile	0.99	Used for $p99(L)$.
Rapid-return threshold δ_{RR}	60 min	Used for ReturnRate_{60}.
Over-cap reference T_{ref}	120 min	Used for OVERCAPMINUTES.
Disparity quantile q	0.2	Top/bottom susceptibility quintiles.
Minimum disparity support	25/group	Otherwise disparity is reported as N/A.
Break prompt default	30 min or 20 items	When enabled, sets $\text{br}_t^+ = 1$ once the threshold is crossed and adds friction $+0.5$.
Session cap default T_{cap}	120 min	When enabled, acts as a hard-stop mitigation wrapper; the final realized item is not truncated.
Throttle default $\lambda_{\max}$	0.6	When enabled, caps personalization at the default throttle value.

B LLM-AUGMENTED BACKEND

B.1 ARCHETYPE-CARD CONSTRUCTION DATA PIPELINE

The default release uses KuaiRec caption/category metadata as the primary semantic source for archetype cards: the raw caption string together with the public first-level and second-level category

fields. KuaiRand supplementary caption/category files are not used in the default card builder. After Unicode normalization and whitespace collapse, each public item is represented by the string

```
caption [SEP] category_level1 [SEP] category_level2.
```

We compute character 3–5-gram TF–IDF vectors from this string, cluster items into exactly Z groups with k-medoids under cosine distance and fixed random seed 0, break clustering ties by smaller medoid item id, sort clusters by descending cluster size, and assign indices $z = 1, \ldots, Z$ in that order. Archetype z is then paired with cluster z, and the medoid item of that cluster provides the fixed semantic anchor for that archetype. The same fixed cluster index is reused wherever appendix diagnostics report content-conditioned targets such as $\mu_r(c)$.

At runtime, $\mathrm{Render}(z_t, \lambda_t, \nu_t, \mathcal{H}_t, \bar{\tau}_t)$ is a pure template function. It does not call the LLM and never generates new free-form card text. The topic line is the normalized medoid caption string for cluster z_t, falling back to the second-level category name and then the first-level category name if the caption is empty. The category line uses the modal $(\mathrm{cat1}, \mathrm{cat2})$ pair within the cluster, with lexicographic tie-breaking. The duration line buckets the medoid-item duration into $[0, 15)$, $[15, 30)$, $[30, 60)$, or $[60, \infty)$ seconds. The popularity line maps the medoid item's empirical interaction-count quintile in KuaiRec to a fixed bucket label. The novelty line is derived from the simulator-visible recent-history count $N_t(z_t)$ of Eq. (11): $N_t(z_t) = 0$ yields "not seen in the last k items," $N_t(z_t) = 1$ yields "seen once in the last k items," and $N_t(z_t) \geq 2$ yields "repeated in the last k items." The intervention line is omitted unless a wrapper is active, in which case it is copied from a fixed wrapper-specific string table. Every field in the card is therefore either public metadata or deterministic bucketization from simulator-visible history.

The language policy is single-language throughout: all released cards, persona strings, and prompt wrappers stay in the original language of the KuaiRec metadata, and the backend uses a multilingual instruction-tuned model. We do not mix languages within a prompt. The example cards below are English glosses for paper readability only.

The synthetic vector x_z used by the parametric core and the public-data semantic anchor attached to CompulsionBench-LLM therefore share the same archetype index, while only the latter is interpreted semantically. The reward-variability flag ν_t is not verbalized directly in the archetype card and continues to affect behavior only through the stochastic multiplicative factor already present in the shared watch-time model.

Example cards. The following are illustrative English glosses of two released card templates; the cached prompts use original-language strings only.

```
Archetype z = 7
Topic tag: home-cooking shortcuts
Category: Food > Recipe
Duration: 15-30 seconds
Popularity: medium
Novelty note: not seen in the last 10 items

Archetype z = 19
Topic tag: stage-dance performance clips
Category: Entertainment > Dance
Duration: 30-60 seconds
Popularity: high
Novelty note: repeated in the last 10 items
Intervention: break reminder active
```

B.2 Prompt schema

Persona template. The pseudo-persona is also deterministic. It is rendered from fixed bucketizations of θ_i into behaviorally descriptive phrases such as "prefers familiar topics," "occasionally explores new topics," "usually stops once sessions get long," and "often keeps browsing once engaged." The renderer never uses diagnostic or pathologizing labels, and it does not generate free-form persona text on the fly.

Watch-phase prompt. Before item consumption, the LLM receives: (i) a pseudo-persona rendered from θ_i; (ii) a compact pre-consumption state summary containing coarse buckets for wall-clock time, current-session minutes, current-session item count, reinforcement trace, control capacity, consumption load, recent repetition, last watch time, and most recent gap; and (iii) the current archetype card. The pseudo-persona is a deterministic rendering of simulator-internal variables and is used only to condition the LLM scorer; it is not treated as an observed public-data annotation. The required JSON output is:

```
{"watch_score": x}
```

with $x \in [0, 1]$. In the notation of Section 3.2, this semantic prior is

$$a_t^{\text{watch}} = f_{\text{LLM}}^{\text{watch}}(\mathcal{P}_i, \mathcal{C}_t, \mathcal{D}_t) \in [0, 1]. \tag{48}$$

Continue-phase prompt. After watch time r_t is realized and the post-consumption state is formed, the LLM receives: (i) the same pseudo-persona; (ii) a compact post-consumption state summary built from $(h_t^+, c_t^+, f_t^+, \tau_t^+, \hat{r}_t^+, \ell_t^+, j_t^+)$; (iii) the same archetype card; and (iv) the realized watch time bucket. The required JSON output is:

```
{"continue_score": y}
```

with $y \in [0, 1]$. The corresponding post-consumption prior is

$$a_t^{\text{cont}} = f_{\text{LLM}}^{\text{cont}}(\mathcal{P}_i, \mathcal{C}_t^+, \mathcal{D}_t, r_t) \in [0, 1]. \tag{49}$$

The canonical watch and continue prompt templates are frozen before second-stage calibration and are reused unchanged for calibration, validity testing, and evaluation. The implementation uses deterministic decoding, exact prompt hashing, strict JSON validation, and response caching. The card builder and persona renderer are separate deterministic preprocessing steps and do not invoke hidden text generation.

B.3 CACHE-FIRST EVALUATION AND LOOKUP TABLES

To make COMPULSIONBENCH-LLM computationally feasible and exactly replayable, we bucket prompt contexts and evaluate the LLM through two discrete lookup tables rather than repeated live inference. The watch-phase cache key is

$$\text{watch_key} = \left(\begin{array}{l} \texttt{persona_bucket, archetype_id, time_bucket,} \\ \texttt{session_bucket, item_count_bucket,} \\ \texttt{repetition_bucket, last_watch_bucket, gap_bucket,} \\ \texttt{lambda_bucket, intervention_bucket} \end{array} \right),$$

and the continue-phase cache key is

$$\text{continue_key} = \left(\begin{array}{l} \texttt{persona_bucket, archetype_id, post_time_bucket,} \\ \texttt{post_session_bucket, post_item_count_bucket,} \\ \texttt{habit_bucket, control_bucket, fatigue_bucket,} \\ \texttt{reward_expectation_bucket, realized_watch_bucket,} \\ \texttt{intervention_bucket} \end{array} \right).$$

Here each bucket is a deterministic quantization of variables already exposed by the fixed prompt renderer: the persona bucket comes from the deterministic pseudo-persona template, $\texttt{archetype_id} = z_t$, the time and post-time buckets are discretizations of $\bar{\tau}_t$ and $\bar{\tau}_t^+$, the session and item-count buckets are discretizations of (ℓ_t, j_t) or (ℓ_t^+, j_t^+), the repetition bucket is derived from $N_t(z_t)$, the habit/control/fatigue/reward-expectation buckets are deterministic discretizations of $(h_t^+, c_t^+, f_t^+, \hat{r}_t^+)$, the realized-watch bucket is a discretization of r_t, and the intervention bucket indexes the active wrapper string or the null wrapper.

We build the watch and continue lookup tables offline by enumerating the unique keys actually visited in the union of calibration rollouts and the fixed main-text evaluation seed rollouts. Each unique key is queried exactly once with deterministic decoding. For every lookup entry we store the bucket key, the raw prompt hash, the raw model output, the parsed scalar score, and a fallback flag indicating

whether the neutral fallback was used after schema repair failed. During benchmark evaluation, the environment performs a table lookup whenever the relevant key is already present; live model calls occur only on cache misses while building or refreshing the cache, never repeatedly inside the reported main-text evaluation loop.

The release must also disclose compute accounting for the cache-first backend: the number of unique watch keys, the number of unique continue keys, the cache hit rate during main-text evaluation, the total number of live model calls required to build the cache, the wall-clock time to build the cache, and the wall-clock time to run the main-text evaluation from that cache. For the cache-first protocol to count as computationally feasible in the main text, the cache hit rate during evaluation must exceed 95%; otherwise the COMPULSIONBENCH-LLM results are treated as appendix-only exploratory analyses.

B.4 SCORE-MAP INITIALIZATION AND SECOND-STAGE CALIBRATION

After fitting the parametric core, we freeze the canonical watch and continue prompt templates together with the released card schema and persona schema before any LLM-specific calibration. Let $\mathcal{C}_{\mathrm{mm}}$ denote the moment-matching context pool, defined as the union of random-policy and PPO rollouts on the held-out calibration split of the calibrated COMPULSIONBENCH-PARAM backend. We query the watch-phase and continue-phase LLM scorers separately on $\mathcal{C}_{\mathrm{mm}}$ and initialize (a_r, b_r, a_c, b_c) by matching the first two moments of raw LLM logits to the corresponding parametric logits. The slope parameters satisfy $b_r > 0$ and $b_c > 0$ throughout calibration in order to preserve monotonicity between the scalar LLM scores and the fused logits. We then optimize only $\Theta_{\mathrm{LLM}} = \{a_r, b_r, a_c, b_c, \omega_{r,\mathrm{llm}}, \omega_{c,\mathrm{llm}}\}$ on the held-out calibration split by minimizing

$$\mathcal{L}_{\mathrm{LLM}}(\Theta_{\mathrm{LLM}}) = \mathcal{L}_{\mathrm{cal}}^{\mathrm{LLM}}(\Theta_{\mathrm{LLM}}) + \lambda_\omega(\omega_{r,\mathrm{llm}}^2 + \omega_{c,\mathrm{llm}}^2), \tag{50}$$

where $\mathcal{L}_{\mathrm{cal}}^{\mathrm{LLM}}$ is the same aggregate public-log target loss as Eq. (10), now evaluated on COMPULSIONBENCH-LLM. This stage is not allowed to tune prompt wording, the card schema, the persona schema, the shared latent-state transition equations, the scorecard, or the task definitions.

B.5 RECOVERY SANITY CHECK

As a backend-consistency test, we evaluate COMPULSIONBENCH-LLM with fusion weights fixed to $\omega_{r,\mathrm{llm}} = \omega_{c,\mathrm{llm}} = 0$ under seed-matched rollouts and verify that it reproduces COMPULSIONBENCH-PARAM exactly up to numerical tolerance. We also report the learned calibrated fusion weights $(\omega_{r,\mathrm{llm}}, \omega_{c,\mathrm{llm}})$ together with the zero-fusion baseline in Table 8. If the fitted fusion weights collapse toward zero, we interpret that as a valid result indicating that the semantic backend adds little beyond the parametric core under the chosen observable calibration targets. This check confirms that COMPULSIONBENCH-LLM is an extension of the same benchmark interface. In the reference implementation, the recovery check bypasses LLM calls entirely when $\omega_{r,\mathrm{llm}} = \omega_{c,\mathrm{llm}} = 0$ and dispatches directly to the COMPULSIONBENCH-PARAM code path, so agreement is exact up to floating-point tolerance.

B.6 AGLLM VALIDITY AND CONTROLLABILITY TESTS

Before treating COMPULSIONBENCH-LLM results as evidence about backend sensitivity, we require a prespecified validity and controllability battery that tests recoverability, output compliance, prompt robustness, directional semantic sanity, and cross-model stability. The primary checkpoint must satisfy every hard threshold in Table 6; otherwise all COMPULSIONBENCH-LLM quantitative comparisons are moved out of the main text and described only as exploratory appendix analyses.

Zero-fusion recovery. The first test is the recovery protocol above. With fusion weights fixed to $\omega_{r,\mathrm{llm}} = \omega_{c,\mathrm{llm}} = 0$ and episode seeds matched exactly, COMPULSIONBENCH-LLM must recover COMPULSIONBENCH-PARAM on the core scorecard up to floating-point tolerance. We report the maximum absolute metric deviation under this setting in Table 8; any non-trivial discrepancy is a hard failure because it would indicate that the LLM backend is not a controlled perturbation of the shared benchmark.

Structured-output compliance. Across all cached watch-phase and continue-phase LLM calls used for calibration and evaluation, we report three schema-compliance statistics: the invalid-JSON rate before repair, the fraction of calls that require the single repair attempt, and the neutral-fallback rate after repair is exhausted. The hard thresholds are an invalid-JSON rate below 0.5% and a neutral-fallback rate below 1%; the repair-attempt rate is reported descriptively because repairs can still yield valid schema-compliant outputs, but unusually large repair rates are disclosed alongside the fallback rate.

Prompt paraphrase invariance. We freeze three semantically equivalent paraphrases of the watch prompt and three semantically equivalent paraphrases of the continue prompt. On $1{,}000$ cached contexts for each phase, we hold the semantic card, pseudo-persona, and state summary fixed and vary only the instruction wording. We then report the mean absolute score deviation from the canonical prompt output, averaged across paraphrases and contexts, and require the worst-case phase-specific deviation to be below 0.03.

Semantic face-validity tests. We construct paired cached contexts that differ in exactly one factor while holding the card, pseudo-persona, and all unrelated state fields fixed. The required pairs increase recent repetition, move the wall-clock later, increase consumption load, lower control capacity, or lengthen the current session. The expected directions are chosen to match the shared benchmark mechanics: more repetition and higher consumption load should weakly reduce watch and continue scores; later wall-clock time and a longer current session should weakly reduce the continue score; and lower control capacity should weakly increase the continue score. We report expected-direction agreement as the fraction of paired comparisons whose score change matches the prespecified sign and require agreement above 80%.

Cross-model sanity check. We evaluate the primary open-weight checkpoint and one stronger sanity-check model on the same cached contexts, using the same seed-matched policy rollouts and the same model-specific calibration protocol. For the fixed policy set used in the paper, we report Kendall's τ for the induced policy ordering on each core metric and require the minimum Kendall's τ across the core metrics to be at least 0.7. This test is not used to redefine the benchmark backend; it is a stability check on whether the reported qualitative ordering is specific to one checkpoint.

B.7 Optional diagnostics

Appendix-only prompt fields such as a post-pause stop preference may be added for exploratory analysis. Because the paper uses only public behavioral data, such fields are not treated as identified constructs and are excluded from the primary scorecard and calibration targets.

The required AGLLM sanity battery combines Table 8 with the validity and controllability thresholds in Table 6; only models that pass that battery are interpreted substantively in the main text.

B.8 AGLLM backend ablations

To isolate what the semantic backend is actually changing, we report four appendix-only backend ablations evaluated on the same fixed policy checkpoints, seed-matched episodes, scorecard, and observable calibration targets used for the main CompulsionBench-LLM comparison. The watch-only variant applies semantic fusion only to the watch score, fixing continuation to the parametric backend by setting $\omega_{c,\mathrm{llm}} = 0$. The continue-only variant applies semantic fusion only to the continuation score, fixing watch-time generation to the parametric backend by setting $\omega_{r,\mathrm{llm}} = 0$. Full CompulsionBench-LLM learns both fusion weights jointly. The non-LLM semantic baseline replaces the LLM with a simple semantic residual model over the same deterministic card and persona inputs, instantiated as sentence-embedding features followed by either a linear head or a small two-layer MLP, and calibrated under the same observable public-log targets as Eq. (50).

This table is intended to answer three questions. First, does policy ranking change? We therefore report the induced ordering of the fixed policy set under each backend and note whether any rank reversals appear relative to CompulsionBench-Param and Full CompulsionBench-LLM. Second, do the learned fusion weights stay small? We therefore report the fitted $\omega_{r,\mathrm{llm}}$ and $\omega_{c,\mathrm{llm}}$ for the LLM ablations and flag cases where either weight becomes large despite weak scorecard

effects. Third, is the LLM doing more than "just another nonlinear perturbation"? We therefore compare Full COMPULSIONBENCH-LLM against the non-LLM semantic baseline on the same core metrics and policy ordering: if the non-LLM residual reproduces the same qualitative ranking and score shifts, the semantic effect is best interpreted as a generic nonlinear perturbation; if Full COMPULSIONBENCH-LLM changes ranking or tail-risk tradeoffs in ways the non-LLM residual does not, then the language-model backend is adding something more specific than a generic semantic smoother.

Fusion family scope. The paper reports only the linear fusion family used in the main text. Richer fusion families and local fusion-weight sensitivity sweeps are not part of the executed appendix and are therefore not claimed as reported results here.

Table 5: Deterministic archetype-card fields.

Card field	Source	Deterministic transform
Topic tag	KuaiRec caption	Cluster-medoid caption after Unicode normalization and whitespace collapse; fallback to category if empty.
Category path	KuaiRec level-1 / level-2 category	Modal category pair within cluster with lexicographic tie-break.
Duration bucket	KuaiRec item duration	Bucket medoid duration into $[0, 15)$, $[15, 30)$, $[30, 60)$, or $[60, \infty)$ seconds.
Popularity bucket	KuaiRec item interaction count	Map medoid interaction-count quintile to a fixed bucket label.
Novelty note	Simulator-visible recent history $\mathcal{H}_t$	Map $N_t(z_t) = 0, 1, \geq 2$ to one of three fixed note strings.
Active intervention text	Wrapper flags and wrapper configuration	Omit when wrappers are off; otherwise insert the corresponding fixed wrapper string.

Table 6: Prespecified AGLLM validity and controllability thresholds.

Test	Reported statistic	Pass threshold	Failure status
Zero-fusion recovery	Maximum absolute deviation from CompulsionBench-Param on the core metrics under $\omega_{r,\mathrm{llm}} = \omega_{c,\mathrm{llm}} = 0$	Exact up to floating-point tolerance	Hard fail
Structured-output compliance	Invalid-JSON rate, repair-attempt rate, neutral-fallback rate	Invalid JSON $< 0.5\%$; fallback $< 1\%$	Hard fail if a threshold is violated
Prompt paraphrase invariance	Mean absolute score deviation across three frozen paraphrases on 1,000 cached contexts per phase	< 0.03	Hard fail
Semantic face-validity	Expected-direction agreement over one-factor paired contexts	$> 80\%$	Hard fail
Cross-model sanity check	Minimum Kendall's τ of policy ordering across the four main-text metrics plus appendix NightMinutes	≥ 0.7	Hard fail

Table 7: Default settings for COMPULSIONBENCH-LLM.

Component	Default	Notes
LLM output fields	`watch_score`, `continue_score`	Both in $[0, 1]$, strict JSON only.
Decoding	deterministic	Temperature 0, fixed seed if supported, cached responses.
Primary checkpoint	`Qwen/Qwen2.5-7B-Instruct`	Open-weight, instruction-tuned, multilingual, and $\leq$ 14B parameters.
Robustness checkpoint	`Qwen/Qwen2.5-14B-Instruct`	Stronger multilingual sanity-check checkpoint used only for cross-model validation.
Tokenizer version	checkpoint-native Qwen tokenizer	Loaded via `transformers==4.46.3`.
Inference library / runtime	`transformers==4.46.3`	PyTorch runtime with deterministic generation and bfloat16 inference.
Quantization setting	none	No quantization in the reference release path.
Max output tokens	16	Strict JSON only; no free-form rationale.
Invalid JSON policy	one repair + neutral fallback	Fallback returns `{"watch_score": 0.5}` or `{"continue_score": 0.5}` as appropriate.
Watch lookup key	bucket tuple	Bucket dimensions: persona, archetype, time-of-day, session minutes, item count, repetition, last watch, gap, lambda, intervention.
Continue lookup key	bucket tuple	Bucket dimensions: persona, archetype, post-time, post-session, post-item-count, reinforcement trace, control capacity, consumption load, reward expectation, realized watch, intervention.
Release manifest	yes	`release/agllm_release_manifest.json` records exact checkpoints, prompt hashes, runtime, and cache-artifact availability.
Released cache artifact	no	This release ships deterministic rebuild scripts rather than precomputed watch/continue tables.
Prompt/card/persona schemas	frozen before calibration	Prompt wording, card fields, and persona buckets are fixed before second-stage calibration and reused unchanged thereafter.
Moment-matching context pool	random-policy + PPO contexts	Drawn from the calibration split of calibrated COMPULSIONBENCH-PARAM under fixed seeds.
Affine maps (a_r, b_r, a_c, b_c)	initialized by moment matching	Matched to parametric logits before aggregate calibration; slope constraints enforce $b_r > 0$ and $b_c > 0$.
Fusion weights $(\omega_{r,\mathrm{llm}}, \omega_{c,\mathrm{llm}})$	learned on calibration split	Set to 0 to recover COMPULSIONBENCH-PARAM.
Fusion family	linear only	Richer fusion families are outside the executed scope of the paper.
Archetype cards	one per z	Deterministic KuaiRec-derived template attached to each benchmark archetype; no card text is generated by the LLM.

Continued on next page

Table 7 (continued)

Component	Default	Notes
Card-builder script	`scripts/build_archetype_cards.py`	Deterministic release script for reconstructing the archetype cards from public metadata and fixed cluster assignments.
Prompt templates	released files	Canonical prompts are released at `prompts/agllm_watch_prompt.txt` and `prompts/agllm_continue_prompt.txt`.
Prompt template hashes	SHA-256 in manifest	Full hashes are recorded in `release/agllm_release_manifest.json`; `watch=91fbd0eb16f6...`, `continue=32f43893da0e....`
Parsing / repair code path	`scripts/parse_agllm_response.py`	Implements strict parse, one repair attempt, and neutral fallback.
Language policy	original-language only	Cards, persona strings, and wrapper text stay in the original KuaiRec language; the LLM is multilingual; no mixed-language prompts.
Main-text use	cache-first evaluation-only extension	Policies are trained in CompulsionBench-Param and re-evaluated in CompulsionBench-LLM from the deterministic cache; main-text cache hit rate must exceed 95%.
Release dashboard stats	reported per release	Invalid JSON rate, repair rate, fallback rate, and cache hit rate are part of the released audit dashboard.

Table 8: LLM backend recovery and calibrated fusion weights.

Setting	$\omega_{r,\mathrm{llm}}$	$\omega_{c,\mathrm{llm}}$	Max. absolute change in CumWatch	Max. absolute change in $\mathrm{CVaR}_{0.95}(L)$
Recovery check	0	0	≈ 0	≈ 0
Calibrated CompulsionBench-LLM	measured	measured	n/a	n/a

Recovery check: LLM calls bypassed; matches CompulsionBench-Param up to floating-point tolerance under seed-matched rollouts.
Calibrated CompulsionBench-LLM: Learned on held-out calibration split; near-zero fitted weights indicate little added semantic signal beyond the parametric core.

C Full Calibration Details

For transparency, we report pre- and post-calibration fits for each target statistic, along with sensitivity to the public-log sessionization threshold Δ_{sess} and to key simulator parameters. This threshold enters only as a post-processing choice for extracting empirical calibration targets; it does not redefine the benchmark's latent simulator sessions or official leaderboard metrics. We treat calibration quality as a property to be measured and disclosed, not assumed. This is especially important because matching aggregate session statistics cannot validate unobserved constructs such as deliberative preference or reinforcement pressure.

The calibration appendix reports the session-length distribution, stopping-hazard curve, inter-session gap distribution, content-cluster watch means, and time-of-day session-start diagnostics for the parametric reference backend. Table 10 summarizes the main interpretation of these diagnostics.

Table 9: Backend ablations for isolating what the semantic backend changes.

Condition	Watch semantic term	Continue semantic term	Calibrated parameters	Interpretation target
Watch-only LLM fusion	learned	parametric only	$(a_r, b_r, \omega_{r,\mathrm{llm}})$ with $\omega_{c,\mathrm{llm}} = 0$	Tests whether semantic effects enter mainly through watch intensity.
Continue-only LLM fusion	parametric only	learned	$(a_c, b_c, \omega_{c,\mathrm{llm}})$ with $\omega_{r,\mathrm{llm}} = 0$	Tests whether semantic effects enter mainly through stopping behavior.
Full COMPULSION BENCH-LLM	learned	learned	$(a_r, b_r, a_c, b_c, \omega_{r,\mathrm{llm}}, \omega_{c,\mathrm{llm}})$	Tests the joint semantic perturbation used in the main text.
Non-LLM semantic baseline	learned residual model	learned residual model	Same observable calibration targets with non-LLM residual weights	Tests whether the effect is more than a generic nonlinear semantic residual.

These are reference-backend diagnostics, not LLM-backend results. LLM calibration plots and recovery diagnostics should be included only for profiles where the validity battery is actually executed.

Table 10: Reference-backend calibration audit summary. The purpose is to disclose fit limits and proxy-activation consequences, not to claim close empirical fit.

Target family	Audit outcome	Consequence for interpretation
Session lengths	Public logs have a longer and heavier session tail than the bounded reference simulator.	Calibration is not external validation; long-session thresholds require activation checks.
Stop hazard	Public target support reaches 9,394 within-session item positions, while the simulator reaches 73 in the reference audit.	Hazard comparison is support-limited; tail fit is a feasibility diagnostic.
Inter-session gaps	Public gaps have much longer support than the simulator gap process.	Rapid-return metrics are in-simulator anti-gaming diagnostics, not calibrated return estimates.
Time-of-day starts	Public logs show a strong diurnal pattern; the simulator is comparatively flat.	Night-use metrics are benchmark proxies, not externally calibrated time-of-day claims.
Content-cluster watch	The current reference fit shows scale and heterogeneity mismatch across clusters.	Cluster-watch calibration is diagnostic; it does not validate item-level realism.

D SUPPLEMENTARY TRAINING, EVALUATION, AND HYPERPARAMETER DETAILS

We provide the full specifications for our data processing, environment calibration, reinforcement learning (RL) training, and compute infrastructure.

D.1 DATASETS AND DATA SPLITS

Unlike standard supervised learning, our benchmark utilizes datasets not as static training splits, but as empirical targets to calibrate the interactive environment and populate the item catalog.

- **Calibration Target Data:** We utilize the full KuaiRand-27K dataset (comprising over 322 million interaction-log rows across approximately 27,000 users) to extract empirical distributions for session lengths, inter-session gaps, watch-time ratios, and stop-hazard support. Sessions are defined using an inactivity threshold of $\Delta_{\mathrm{sess}} = 30$ minutes.
- **Semantic Content Catalog:** We utilize KuaiRec to populate the environment's item catalog. Specifically, we extract the `item_id`, `duration_sec`, and `caption` fields, merging

them with aggregated interaction counts to construct the metadata-backed cards required by the COMPULSIONBENCH-LLM backend.

D.2 ENVIRONMENT CALIBRATION AND HYPERPARAMETERS

The baseline parametric environment (COMPULSIONBENCH-PARAM) is calibrated using the Optuna hyperparameter optimization framework via Tree-structured Parzen Estimator (TPE) sampling. The objective is to minimize the distributional distance between the simulated rollouts and the KuaiRand-27K empirical logs.

To ensure stability across the massive search space, we utilize a multi-stage funnel approach:

1. **Exploratory Search:** 300 trials, evaluated over 200 episodes each.
2. **Candidate Selection:** The top 20 candidates from the exploratory search are re-evaluated over 1,000 episodes.
3. **Finalist Audit:** The top 5 candidates are rigorously evaluated over 5,000 episodes, and the best-performing configuration is frozen.

The final frozen configuration, which is used identically across all RL experiments, includes: load smoothing $\rho_f = 0.08$, control depletion $\chi_c = 0.25$, reinforcement-trace prior median $\rho_{h,\mathrm{loc}} = -3.0$, and control-capacity prior median $\rho_{c,\mathrm{loc}} = -2.5$. The complete parameter dictionary is provided in the repository as `config_calibrated.json`.

D.3 REINFORCEMENT LEARNING DETAILS

The benchmark's reference policies are trained using Proximal Policy Optimization (PPO), implemented via Stable-Baselines3.

- **Optimizer:** Adam with default Stable-Baselines3 learning rate (3×10^{-4}).
- **Rollout buffer:** 2,048 steps per rollout.
- **Training Budget:** Each policy is trained for a total of 1,000,000 environment steps to ensure policy convergence.
- **Statistical Robustness:** To account for stochasticity in RL exploration, all policies are trained across 5 independent random seeds (seeds 0–4). Results are aggregated across seeds for the final scorecard.

D.4 EVALUATION METHODOLOGY

Because compulsion-risk proxies are heavily skewed (i.e., long-tailed binge behavior), standard short-horizon evaluations yield highly noisy metric estimates. Therefore, the final trained policies are evaluated over 5,000 episodes per seed. This high evaluation budget is strictly enforced to accurately estimate tail risk metrics, such as the 95th-percentile Conditional Value at Risk ($\mathrm{CVaR}_{0.95}$) for over-cap minutes.

D.5 COMPUTE INFRASTRUCTURE AND WALL-CLOCK TIMES

All data preprocessing, Optuna calibration, and PPO training pipelines were executed on a Community Cluster which features Dell compute nodes equipped with Intel Xeon processors and high-speed 100 Gbps Infiniband interconnects.

Specifically, the parallelized array jobs utilized the `a100-80gb` partition, allocating 1 NVIDIA A100 (80GB) GPU and 8 CPU cores per training seed. Due to the massive scale of the KuaiRand-27K dataset, Pandas memory overhead during the calibration and evaluation phases routinely exceeded 170 GB; thus, compute nodes were provisioned with 240 GB of RAM. The COMPULSIONBENCH-LLM second-stage fit and evaluation utilized local inference via Qwen2.5-7B-Instruct, loaded directly onto the A100 VRAM to avoid external API latency.

Compute Time Accounting: The total computational budget for this study was approximately 130 GPU hours. The Optuna calibration funnel required ~ 4 wall-clock hours on a single A100. For the parametric baseline (COMPULSIONBENCH-PARAM), training the policies for 1,000,000 steps across 5 seeds took ~ 12 wall-clock hours per seed. For the hybrid model (COMPULSIONBENCH-LLM), generating the initial semantic proxy cache via local Qwen2.5-7B inference required ~ 6 wall-clock hours, followed by ~ 14 wall-clock hours per seed for the subsequent RL training and evaluation loop.

D.6 ADDITIONAL BASELINE AND REPRODUCIBILITY SPECIFICATIONS

Offline data for the myopic baseline. The myopic ranker is trained on a logged dataset of 100,000 transitions collected from COMPULSIONBENCH-PARAM-DEFAULT under a uniformly random policy. The ranker uses the same observation representation as the learned policies and is trained to predict one-step watch time for candidate actions.

Policy architectures and training budgets. All learned policies use the same small MLP backbone. The active-threshold run uses multiple train seeds and held-out test episodes per seed for the official scorecards; the 120-minute reference run uses its own released seed list and held-out evaluation episodes. Exact optimizer settings, training budgets, model-selection rules, and seed lists are reported in the release documentation.

Model selection and constraint operating points. For each training seed, unconstrained methods select the checkpoint with highest validation reward. Constrained candidates are screened for feasibility when possible, and least-violating fallbacks are explicitly logged when no feasible checkpoint exists at a search point. This matters for interpretation: Lagrangian PPO is reported as a constrained baseline and sensitivity analysis rather than as a clean proof that constrained RL reliably recovers a feasible frontier.

Evaluation episode counts and reproducibility assets. The policy evaluation uses the fixed validation and test seed lists released in the manifest; in the saved parametric bundle these correspond to 1,000 validation episodes and 500 test episodes for the main frontier and scorecard tables, while some rebuild-only diagnostics use smaller explicitly documented slices. The appendix AGLLM episode counts specify the gating protocol; this policy-evaluation bundle does not include regenerated backend comparisons. The public release includes the default configuration together with its calibration manifest and audit table, the stress-test and ablation variants, deterministic evaluation scripts, seed lists, released result summaries, and scripts for regenerating tables and plots from those summaries. Unrestricted PPO checkpoints are not part of the public release.

E FULL METRIC DEFINITIONS

This appendix collects the full formal metric sheet used by the benchmark. Let L denote session length in minutes, G denote the gap between two consecutive sessions, S_t denote cumulative watch time in the current session before step t, $\bar{r}_t$ denote wall-clock time at the start of step t, and r_t denote watch time accrued during that step. Throughout this metric sheet, sessions are the benchmark's latent simulator sessions rather than inactivity-threshold sessions from post-processing.

E.1 CORE SCORECARD

The core scorecard consists of

$$\text{CUMWATCH} = \mathbb{E}\left[\sum_t r_t\right], \tag{51}$$

$$\text{CVaR}_{0.95}(L) = \mathbb{E}[L \mid L \geq \text{VaR}_{0.95}(L)], \tag{52}$$

$$\text{OVERCAPMINUTES} = \mathbb{E}\left[\sum_t e_t\right], \qquad e_t = \big(S_t + r_t - T_{\text{ref}}\big)_+ - \big(S_t - T_{\text{ref}}\big)_+. \tag{53}$$

When promoted by the release's proxy-orthogonality audit, the main scorecard additionally reports a night-use proxy such as LATENIGHTSESSIONSTARTRATE or NIGHTFRACTION. For horizon-censored terminal sessions, $\mathrm{CVaR}_{0.95}(L)$ uses the realized latent session length accrued up to the episode horizon, rapid-return metrics exclude the terminal session because its post-session gap is unobserved, and OVERCAPMINUTES counts only the realized pre-horizon watch time accrued before censoring.

E.2 EXTENDED SCORECARD ITEMS

For appendix analyses, we additionally report

$$\textsc{NightMinutes} = \mathbb{E}\left[\sum_t \mathrm{Overlap}([\bar{\tau}_t, \bar{\tau}_t + r_t), \mathcal{W}_{\mathrm{night}})\right], \tag{54}$$

$$\textsc{NightFraction} = \frac{\textsc{NightMinutes}}{\max(\textsc{CumWatch}, \epsilon)}, \tag{55}$$

$$\textsc{LateNightSessionStartRate} = \Pr(\text{session start} \in \mathcal{W}_{\mathrm{night}}), \tag{56}$$

$$\mathrm{ReturnRate}_{60} = \Pr(G \le 60 \mid G \text{ observed}), \tag{57}$$

$$\mathrm{p99}(L) = \inf\{\ell : \Pr(L \le \ell) \ge 0.99\}. \tag{58}$$

Subgroup disparity analyses use simulated susceptibility bins and should be interpreted as benchmark-internal stress tests rather than real-world fairness audits over protected attributes.

E.3 HAZARD PLOTS AND LATENT DIAGNOSTICS

Stopping-hazard plots are reported as

$$h_{\mathrm{stop}}(j) = \Pr(\text{stop at within-session position } j \mid \text{session reaches } j),$$

and return-hazard plots summarize the empirical hazard of the inter-session gap distribution over observed gaps. Latent diagnostics include

$$\mathrm{OverrideRate} = \mathbb{E}\Big[\mathbb{I}[\text{continue}] \cdot \mathbb{I}[Q_t^{\mathrm{M},+} < 0]\Big], \tag{59}$$

where $Q_t^{\mathrm{M},+}$ is the simulator's latent deliberative/outside-option component evaluated on the same post-consumption state used for the realized continue/stop draw in Appendix A, together with

$$\mathrm{AvgHabit} = \mathbb{E}[h_t], \qquad \mathrm{CtrlDepletion} = \mathbb{E}[1 - c_t]. \tag{60}$$

When the break-prompt wrapper is enabled and at least one prompt-triggering state is observed, we additionally report

$$\textsc{BreakAdherence} = \Pr(y_t = \mathrm{stop} \mid \mathrm{br}_t^+ = 1) = \frac{\mathbb{E}[\mathbb{I}[y_t = \mathrm{stop}]\,\mathbb{I}[\mathrm{br}_t^+ = 1]]}{\mathbb{E}[\mathbb{I}[\mathrm{br}_t^+ = 1]]}, \tag{61}$$

the fraction of prompt-triggering states that are followed by stop rather than continue. The appendix dashboard always includes the stopping-hazard plots above, and includes BREAKADHERENCE only when the break wrapper is active. These quantities are reported only as post-hoc diagnostics and are not part of the main text scorecard.

F STRESS TESTS AND ABLATIONS

F.1 STRESS-TEST DEFINITIONS

To assess robustness beyond the default configuration, we evaluate trained policies under one-factor perturbations such as susceptibility shifts, slower recovery, and higher reward variability. These perturbations are structured robustness checks around the reference benchmark rather than new datasets. Sensitivity to the public-log sessionization threshold Δ_{sess} is reported separately as a calibration post-processing analysis rather than as an alternative benchmark session definition.

F.2 Structural component ablations

The executed component-ablation bundle includes NoHabit, NoVar, NoBeliefMixing (NoPers in the implementation), and HomogeneousUsers, evaluated for PPO and Lagrangian PPO on the same held-out evaluation seeds used for the active-threshold release. Latent diagnostics and component ablations help interpret why a policy changes observable outcomes, but they are not benchmark tracks and are not used to define the primary comparisons.

F.3 Appendix wrapper baselines

The executed appendix wrapper baselines are break prompts, autoplay-off, and personalization throttles in addition to the session cap used in the main text. These appendix comparisons are intended as robustness checks rather than as part of the core scientific comparison, and their generated summaries are emitted in `tables/table_appendix_wrapper_baselines.csv`.

F.4 Cap-threshold sensitivity

To assess whether the post-hoc session cap appears strong only because the default cap threshold matches the scorecard threshold, we fix T_{ref} and vary the intervention threshold T_{cap} over representative values near that anchor (e.g., $\{10, 15, 20\}$ or $\{90, 120, 150\}$, or quantiles $q \in \{0.9, 0.95, 0.99\}$). We evaluate the post-hoc cap applied to the learned PPO policy and report cumulative watch time, $\mathrm{CVaR}_{0.95}(L)$, $\mathrm{ReturnRate}_{60}$, OverCapMinutes, and the session-cap trigger rate. This sensitivity check is intended to show whether the cap baseline also changes non-optimized outcomes such as tail session length and rapid re-entry when the intervention threshold is decoupled from the scorecard threshold. In the reference 120-minute profile, the cap grid is inactive, which serves as a proxy-activation sanity check to ensure the cap does not trigger in settings where it is not intended to act.

G Additional Results

G.1 Stochastic companion scorecard

The stochastic scorecard is reported separately from the deterministic main comparison. It checks whether the mitigation conclusions survive rollout stochasticity rather than redefining the official deterministic ordering.

Table 11: Stochastic companion scorecard for the active-threshold profile. Values are means ± episode-bootstrap 95% confidence intervals.

Policy	Watch ↑	CVaR ↓	OverCap ↓	NightStart ↓
Random	75.74 ± 1.06	21.02 ± 0.35	4.13 ± 0.26	0.251 ± 0.004
Myopic	75.54 ± 1.08	21.12 ± 0.31	4.20 ± 0.23	0.254 ± 0.004
PPO	75.28 ± 1.00	20.98 ± 0.34	4.09 ± 0.25	0.251 ± 0.004
Lagrangian PPO	74.56 ± 0.99	20.85 ± 0.34	3.96 ± 0.25	0.250 ± 0.005
PPO + SessionCap(15)	71.96 ± 0.94	15.73 ± 0.04	0.50 ± 0.03	0.250 ± 0.004

G.2 Cap incidence, severity, and anti-fragmentation diagnostics

Table 12 reports the episode-level diagnostics behind the main mitigation finding. The cap leaves threshold-crossing incidence almost unchanged but sharply reduces conditional and tail severity. Short-gap return rates remain near zero.

G.3 Training convergence and PPO optimization

Figure 2 summarizes the training traces released with the active-threshold profile. We include this diagnostic because the benchmark is intended for reproducible evaluation: scorecard comparisons should be accompanied by training curves and checkpoint-selection rules, not only final numbers.

Table 12: Episode-level cap diagnostics. Pos. is the fraction of episodes with positive over-cap exposure; p95 and p99 are episode-level quantiles of OVERCAPMINUTES; Ret5 is the mean fraction of observed returns within five minutes.

Mode	Policy	Pos.	Mean	p95	p99	Ret5
Det.	PPO	44.1%	2.84	14.23	24.90	0.02%
Det.	PPO + Cap	44.1%	0.45	2.36	4.27	0.02%
Stoch.	PPO	55.2%	4.09	18.55	28.83	0.04%
Stoch.	PPO + Cap	55.5%	0.50	2.11	3.65	0.02%

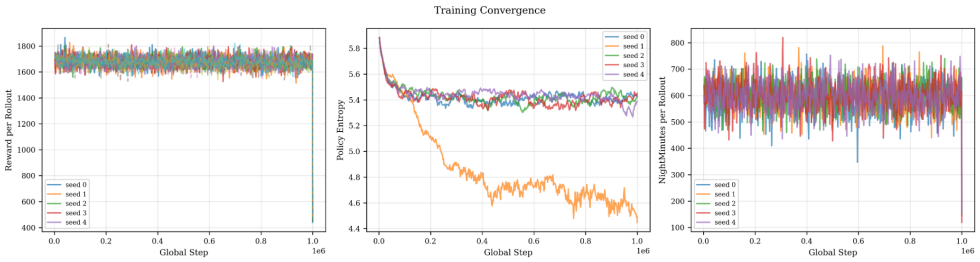

Figure 2: Training convergence diagnostics for the released PPO and Lagrangian PPO runs. The figure is an audit artifact for optimization progress and checkpoint selection rather than a separate benchmark score.

G.4 PROXY-ORTHOGONALITY AUDIT

The active-threshold profile rejects raw NIGHTMINUTES as a main proxy because it is nearly collinear with cumulative watch time. The audit promotes LATENIGHTSESSIONSTARTRATE for descriptive reporting because its policy-mean correlation with watch time is lower and it changes policy comparisons.

Table 13: Night-proxy orthogonality audit for the active-threshold profile.

Candidate metric	Watch corr.	Comparison changes?	Main-text status
NIGHTMINUTES	0.999	No	Rejected as scalar proxy
NIGHTFRACTION	0.819	Yes	Appendix diagnostic
LATENIGHTSESSION-STARTRATE	0.757	Yes	Promoted descriptive proxy

H BENCHMARK CARD

Intended use. COMPULSIONBENCH is intended for research on comparative auditing of recommendation policies under explicit simulator assumptions. The benchmark is designed to test whether policies that increase engagement also change observable long-horizon risk proxies such as prolonged sessions, rapid re-entry, night-time use, and over-cap exposure. Appropriate uses include benchmarking RL and bandit-style recommenders, comparing constrained versus unconstrained training, and evaluating abstract mitigation wrappers such as session caps, autoplay-off, or throttled personalization.

Supported tasks. The two official benchmark tasks are: (i) engagement maximization, measured by cumulative watch time together with the released main-text risk scorecard, and (ii) observable-risk constrained optimization, where policies maximize watch time subject to the OVERCAPMINUTES constraint used in the current release. Appendix analyses include the full proxy-orthogonality audit, rapid-return sensitivity tables, stress tests, the executed component-ablation bundle, wrapper baselines, and latent diagnostics; these are not separate leaderboard tracks.

Non-goals. COMPULSIONBENCH is not a clinical instrument, not a prevalence estimator, not a validated model of user psychology, and not deployment guidance for live recommender systems. The benchmark is not intended to certify that a policy is safe, beneficial, or acceptable for real users. It is also not intended to replace human-subject evaluation, field studies, or external audits of deployed products.

Core metrics. Official reporting centers on three main-text observable metrics computed from latent simulator sessions: CUMWATCH, $\mathrm{CVaR}_{0.95}(L)$, and OVERCAPMINUTES. The appendix dashboard additionally surfaces the full proxy-orthogonality audit, appendix-only night-family metrics, $\mathrm{ReturnRate}_{60}$ and $\mathrm{ReturnRate}_{5}$ sensitivity tables, $p99(L)$, BREAKADHERENCE when break prompts are enabled, and stopping-hazard plots. Latent diagnostics such as OVERRIDERATE, AVGHABIT, and CTRLDEPLETION are appendix-only interpretation tools and are not benchmark targets.

Known validity limits. The environment is synthetic and uses aggregate public-log statistics only as calibration targets rather than as individual-user fits. Latent variables such as the reinforcement trace, control capacity, consumption load, and deliberative/outside-option score are modeling constructs, not validated measurements of real cognition or pathology. Public-log sessionization uses an inactivity threshold Δ_{sess} only for calibration target extraction; official benchmark metrics use the simulator's latent session boundaries. Released calibration audits improve transparency about where the fit is weak, but they do not justify causal claims, clinical claims, or external prevalence estimates. The intervention wrappers are intentionally abstract and should not be interpreted as full user-interface or policy-deployment designs. Any subgroup disparity analysis over simulated susceptibility bins should be interpreted as a benchmark-internal stress test rather than as a real-world fairness audit over protected attributes.

Released assets. The public release includes the default environment together with calibration manifests and audit tables, configuration files, stress-test and ablation variants, deterministic evaluation scripts, and released result summaries sufficient to reproduce the reported scorecard tables and plots. The release also documents seeds, metric definitions, checkpoint-selection rules, and failure-mode assumptions. Unrestricted high-engagement PPO checkpoints are not included in the public release. The release also includes the optional COMPULSIONBENCH-LLM extension, which shares the same interface and metrics but swaps in the LLM-augmented response backend together with exact checkpoint identifiers, the release manifest at `release/agllm_release_manifest.json`, prompt templates at `prompts/agllm_watch_prompt.txt` and `prompts/agllm_continue_prompt.txt`, the deterministic card-builder script at `scripts/build_archetype_cards.py`, the parsing/repair path at `scripts/parse_agllm_response.py`, bucket definitions, prompt hashes, and scripts to rebuild the watch/continue caches locally from the released open-weight models. The released audit dashboard also reports the invalid-JSON rate, repair rate, fallback rate, cache-hit rate, and the public-log calibration fit status for the exact release configuration.

Expected reporting format. Any benchmark report should identify the task, policy class, training budget, checkpoint-selection rule, evaluation episode counts, and whether the result is from the default benchmark, an appendix stress test, or an appendix-only extension. Reports should present mean and 95% confidence intervals for the three released main-text scorecard metrics on COMPULSIONBENCH-PARAM-DEFAULT, and should disclose the proxy-orthogonality audit whenever night-family metrics are discussed. If the COMPULSIONBENCH-LLM extension is used, the same main-text metrics should also be reported on the paired COMPULSIONBENCH-LLM-DEFAULT evaluation using the same policy checkpoint and seed list. Reports using COMPULSIONBENCH-LLM should additionally disclose the exact checkpoint identifiers, tokenizer/runtime/quantization settings, prompt template hashes, invalid-JSON rate, repair rate, fallback rate, and cache-hit rate for the evaluated release configuration. If latent diagnostics are included, they should be labeled explicitly as post-hoc diagnostics rather than headline benchmark scores. Reports using COMPULSIONBENCH-LLM should explicitly state whether policies were trained in COMPULSIONBENCH-PARAM and only re-evaluated in COMPULSIONBENCH-LLM, or fine-tuned in COMPULSIONBENCH-LLM as an appendix-only extension.

I MISUSE STATEMENT

Harmful-engagement risk. Because COMPULSIONBENCH can be used to search for policies that maximize watch time under long-horizon user dynamics, it could be misused as a recipe for harmful engagement optimization. The paper's motivating concern is precisely that unconstrained engagement optimization may increase observable risk proxies. Benchmark use should therefore be framed as auditing and comparison under explicit assumptions, not as a target for product optimization.

Reporting prohibitions. Results from COMPULSIONBENCH should not be reported as watch time alone. Any claim about a policy's performance must pair CUMWATCH with the benchmark's released observable risk metrics, at minimum $\mathrm{CVaR}_{0.95}(L)$, OVERCAPMINUTES, and any night proxy explicitly promoted by the release's proxy-orthogonality audit; rapid-return metrics should be labeled as appendix sensitivity reporting in the current release. Latent diagnostics do not substitute for the required observable scorecard, and appendix-only extensions should be labeled as such.

Real-world and clinical claims. Benchmark scores must not be treated as estimates of real-world prevalence, public-health burden, addiction incidence, or clinical evidence of harm. They must not be presented as evidence that a deployed platform causes or avoids addiction. The benchmark uses aggregate public-log calibration targets, released fit audits, and non-clinical latent constructs; its outputs support within-benchmark comparisons only.

Checkpoint release stance. Unrestricted PPO checkpoints will not be released in the public benchmark package. If any high-engagement unconstrained checkpoints are later shared for tightly scoped reproducibility work, they should be labeled research-only, excluded from deployment use, and accompanied by the full paired benchmark scorecard rather than by watch-time numbers alone.