PREDICTIVE TRAJECTORY WORLD MODELS FOR PROCESS RISK IN LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

An LLM agent can reach a correct outcome while following an unreliable execution process. Outcome-level evaluation can therefore miss process failures, while fixed anomaly taxonomies provide task- and dataset-dependent definitions of abnormal behavior. Instead of defining abnormal agent behavior with a fixed taxonomy, we model normal execution and measure how a candidate action is predicted to depart from it. Given a completed trajectory prefix and a candidate next action, we study prospective process risk by predicting before execution whether that action will move the process outside its normal support. Our analysis finds cross-model consistency for normal executions of the same exact task but limited evidence that a whole task category forms one compact process cluster. This motivates task-conditioned local normal support. This also separates executor-specific variation from task information, which remains necessary to define what constitutes normal execution. We align same-task executions across executor models and use retrieved normal context in a trajectory world model that predicts the next latent process state under the candidate action. The world model learns normal process dynamics rather than directly predicting anomaly labels, leaving anomaly supervision to the final onset-aware risk mapping. Predicted future-state and transition deviations then inform an onset-aware risk score focused on the first abnormal step. Experiments support each part of this chain. Geometry analysis shows clear same-task cross-model structure, but does not support a single compact center for a whole task category. Retrieval-conditioned prediction achieves the lowest mean next-state error across cross-model, held-exact-task, and leave-one-subset-out settings. At the 10% development normal-trajectory FAR budget, the final predictive-risk model reaches 56.8% first-alarm exact-onset hit, improving over the matched direct-only baseline by 17.3 percentage points at similar observed FAR. These results show that process risk can be grounded in task-conditioned normal structure and predicted future deviation.

1 INTRODUCTION

LLM agents act through sequences of reasoning, actions, and environment observations (Yao et al., 2023). A correct final outcome therefore does not by itself establish that the execution process was reliable. Studies of mathematical reasoning distinguish final-answer correctness from errors in intermediate reasoning (Uesato et al., 2022; Lightman et al., 2024). Tool-using agents raise a further concern: intermediate actions can affect external state and expose safety or security risks (Ruan et al., 2024; Debenedetti et al., 2024). An agent may act on weak evidence, choose an inappropriate tool call, or create an external effect. It may still achieve the task goal even when its execution process contains abnormal intermediate behavior. Such abnormal process may remain invisible under final-outcome evaluation. OpenClawBench characterizes this mismatch as the outcome–process gap, showing that successful task completion can still involve abnormal execution processes (Liu et al., 2026). This motivates evaluating the process itself, not only its final outcome.

Existing studies define problematic behavior under different taxonomies, including unsafe tool use, prompt injection, and other task-specific execution risks (Ruan et al., 2024; Debenedetti et al., 2024; Xie et al., 2025; Xia et al., 2025). These annotations are useful for identifying known failure modes, but their definitions and boundaries vary across tasks, datasets, and base models. A fixed taxonomy does not provide a common reference for distinguishing reliable from abnormal execution. We

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

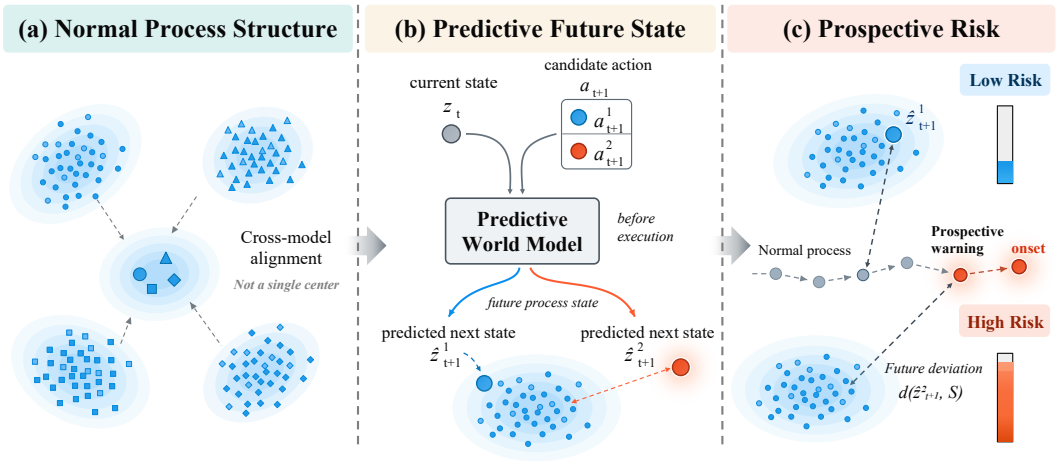


Figure 1: Overview of the proposed framework. (a) normal agent execution is modeled as task-conditioned local process structure, with same-task executions aligned across executor models. (b) A retrieval-conditioned trajectory world model uses the current trajectory, task context, and candidate action to predict the next process state. (c) State and transition deviations from local normal support are combined into an onset-aware risk score for identifying the first abnormal transition.

instead start from normal execution and study whether normal trajectories exhibit stable process. Abnormal behavior can then be characterized by its departure from normal process structure rather than only by membership in predefined failure categories. Modeling normal execution is therefore necessary to define abnormality through process deviation.

A useful representation of normal execution must separate executor variation from information that determines what behavior is appropriate. Agents built on different backbone models may follow different execution styles, while the task goal and execution stage constrain which actions are normal. Our goal is to obtain a process representation that is robust across executor models while preserving task semantics and ordered trajectory information. Our geometry analysis shows that normal executions of the same task are consistent across backbone models. However, we find much weaker support for a single compact cluster across tasks in the same category. This suggests that normality should not be represented by a category-wide center. Instead, we use task-conditioned local normal support that preserves task- and stage-specific structure while reducing executor-specific variation.

Modeling normal execution alone is not enough to assess the risk of a candidate action. A trajectory prefix may currently lie within normal support, while different candidate actions can lead to different next states. Static current-state scoring cannot capture the consequence of the action that has not yet been executed. We address this by learning a trajectory world model that predicts the next process state from the current trajectory, task context, and candidate action. Retrieved normal context provides the local process conditions under which the transition should be interpreted. In this sense, the world model learns action-conditioned process dynamics rather than directly predicting anomaly labels (Ha & Schmidhuber, 2018; Hafner et al., 2020). Predicting the next state makes it possible to evaluate where the candidate action is likely to move the execution before that action is executed.

Once the next process state can be predicted, risk can be defined by how far that future state departs from the normal support. A candidate action is low-risk when its predicted next state and transition remain compatible with normal local behavior, and high-risk when they move outside that support. We combine predicted future-state deviation, transition deviation, and local support statistics into an onset-aware risk score. Supervision focuses on the first abnormal transition: pre-onset steps remain negative, the onset step is the target event, and post-onset states are not part of the primary early-warning objective. This formulation treats process risk as prospective deviation rather than current abnormality. It also makes the role of the world model explicit: risk is determined by the predicted consequence of the candidate action, not only by the action text or the current trajectory state.

The main contributions of this work are as follows:

- **normal process representation.** We model normal agent execution as task-conditioned and locally structured. normal executions of the same task are consistent across executor models, while task and execution-stage information are preserved.
- **Predictive trajectory world model.** We develop a retrieval-conditioned trajectory world model. It predicts the next process state from the current trajectory task context and candidate action, and generalizes across executors and unseen tasks.
- **Prospective process risk.** We define onset-aware process risk from the predicted departure from task-conditioned normal process support. State and transition deviation under candidate action are then used to identify the first abnormal transition before execution.

2 Related Work

2.1 Trajectory Anomaly Detection and Process Auditing

Recent work has moved agent evaluation from final outcomes to the execution process. OpenClaw-Bench shows that successful task completion can still contain process-side anomalies, and provides trajectory-level labels together with the first abnormal onset for real agent executions (Liu et al., 2026). Building on this setting, our work studies whether a candidate action will lead the trajectory into an abnormal state. Existing studies usually define abnormal behavior through predefined taxonomies or annotated risk categories. ToolEmu evaluates agent failures in curated high-risk scenarios, R-Judge annotates interaction records with safety labels and risk types, and Agent-SafetyBench organizes unsafe behavior into predefined risk categories and failure modes (Ruan et al., 2024; Yuan et al., 2024; Zhang et al., 2024). Other benchmarks focus on specific threats, such as prompt injection in AgentDojo and InjecAgent, or unsafe tool use in ToolSafety and SafeToolBench (Debenedetti et al., 2024; Zhan et al., 2024; Xie et al., 2025; Xia et al., 2025). ToolSafe and TRACES further support risk detection during agent execution, but still rely on predefined safety supervision or trajectory-level risk labels (Mou et al., 2026; Li et al., 2026). These definitions are useful for known failure modes, but their boundaries can vary across tasks, datasets, and base models. A fixed taxonomy therefore provides no common reference for abnormal execution. We instead model normal process structure and estimate risk from the predicted deviation caused by a candidate next action.

2.2 Predictive Process Representation and World Models

Agent execution is sequential, and the effect of an action depends on the preceding trajectory (Yao et al., 2023). World models learn action-conditioned state dynamics and have been widely used for prediction, planning, and control. PlaNet and Dreamer learn latent dynamics for future prediction and imagined rollouts, while MuZero learns task-relevant dynamics directly for planning (Hafner et al., 2019; 2020; Schrittwieser et al., 2020). Recent LLM-agent studies extend this idea to interactive environments. WorldCoder builds executable world models from agent experience, and WALL-E aligns an LLM-based world model with environment dynamics for look-ahead action selection (Tang et al., 2024; Zhou et al., 2025). These methods mainly use world models for environment simulation or long-horizon planning. Our goal is to use the world model to predict the next process state induced by a candidate action, and then measure its deviation from task-conditioned normal support. This makes predicted process dynamics an explicit signal for risk estimation.

3 Problem Formulation

We formalize prospective process-risk prediction for ReAct-style agent trajectories. The key question is whether a candidate next action will cause the execution process to enter its first abnormal state from the current completed trajectory prefix. We first define the agent trajectory and the pre-execution information boundary, and then specify the onset-aware target used for risk prediction.

3.1 Agent Trajectory

Let an execution of task q be $\tau = (q, e_1, \ldots, e_T)$, $e_i = (\rho_i, a_i, o_i)$ where ρ_i, a_i, and o_i denote the reasoning, action, and observation at each step i, respectively. At decision time t, the completed

prefix is $P_t = (q, e_1, \ldots, e_t)$, with $P_0 = (q)$, and a_{t+1} is the proposed next action. The auditor estimates the process risk of the candidate next action before execution:

$$r_{t+1} = \mathcal{R}(P_t, a_{t+1}), \qquad o_{t+1} \notin \text{input}(\mathcal{R}). \tag{1}$$

Because a_{t+1} has not been executed, o_{t+1} and future trajectory information are unavailable. Source, subset, anomaly, and onset identifiers are not runtime forward inputs. The objective is to determine whether the candidate action will initiate an abnormal process transition.

3.2 Process-Risk Prediction

For an anomalous trajectory, let k be the first annotated abnormal step. The candidate-step target is

$$y_i = \begin{cases} 0, & i < k, \\ 1, & i = k, \\ \text{excluded}, & i > k. \end{cases} \tag{2}$$

Pre-onset steps are negative because the abnormal transition has not yet begun; the onset step is positive because it initiates that transition. Post-onset steps are excluded because the process is already abnormal at that point. Every eligible step of a normal trajectory is negative. The score $r_{t+1} \in [0, 1]$ is used as a ranking and alarm score, not necessarily a calibrated probability.

4 Method

We develop a predictive trajectory world model for estimating whether a candidate next action will trigger the first abnormal process transition. The method models task-conditioned normal process structure, predicts the next process state under the candidate action, and combines direct task–action evidence with predicted-process evidence for onset-aware risk estimation.

4.1 Method Overview

Figure 2 summarizes the full pipeline as two complementary evidence paths that meet at onset-aware risk fusion. We write the overall computation as

$$\begin{aligned} h_t &= E_\theta(P_t), \\ \mathcal{N}_t &= \text{Retrieve}(q, a_{t+1}, \mathcal{B}_N), \\ (\hat{z}_{t+1}, u_{t+1}) &= F_\phi(h_t, a_{t+1}, q, \mathcal{N}_t), \\ v_{t+1}^{\text{dir}} &= G_{\text{dir}}(q, a_{t+1}), \\ v_{t+1}^{\text{pred}} &= G_{\text{pred}}(h_t, \hat{z}_{t+1}, \mathcal{N}_t, u_{t+1}), \\ r_{t+1} &= H_\psi(v_{t+1}^{\text{dir}}, v_{t+1}^{\text{pred}}). \end{aligned} \tag{3}$$

Here h_t denotes the representation of the completed process, and $\mathcal{N}_t$ is the local normal support retrieved for the current task and candidate action. The world model predicts the next process state $\hat{z}_{t+1}$ together with its uncertainty u_{t+1}. The direct view v_{t+1}^{dir} captures task–action evidence available before execution, while v_{t+1}^{pred} summarizes evidence from the predicted process dynamics, local normal support, and uncertainty. Their onset-aware fusion produces the final process-risk score r_{t+1}. Figure 2 illustrates these two evidence paths and the resulting first-alarm decision.

4.2 normal Process Modeling

normal execution depends on the task and its current stage. We summarize the ordered events as

$$h_t = E_\theta(e_1, \ldots, e_t). \tag{4}$$

Here, e_i denotes the i-th completed ReAct event, and E_θ encodes the ordered event sequence into the current process representation h_t. Thus, h_t summarizes the observed execution history up to step t. Task semantics are retained as an explicit conditioning signal for retrieval, prediction, and risk estimation rather than being treated as nuisance variation.

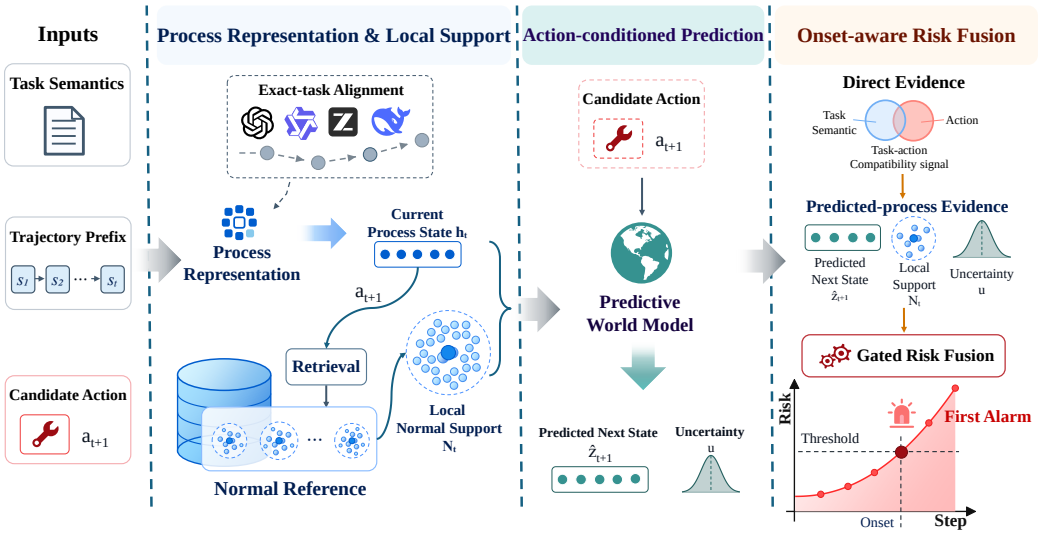


Figure 2: Overview of our predictive trajectory world model for prospective process risk. Given task semantics, an observed trajectory prefix, and a candidate next action, the method builds task-conditioned local normal support and predicts the next process state before execution. Direct task–action evidence and predicted-process evidence are then fused into an onset-aware risk score, whose first threshold crossing triggers the first alarm.

For exact task g, let $\mathcal{T}_g^{(m)}$ be its fit-side normal executions under executor m, and let $\mathcal{S}_g$ contain the executors with such executions. We align their terminal process representations:

$$\begin{aligned} c_g^{(m)} &= \frac{1}{|\mathcal{T}_g^{(m)}|} \sum_{\tau \in \mathcal{T}_g^{(m)}} h_T(\tau), \\ \bar{c}_g &= \frac{1}{|\mathcal{S}_g|} \sum_{m \in \mathcal{S}_g} c_g^{(m)}, \\ \mathcal{L}_{\text{align}} &= \sum_g \sum_{m \in \mathcal{S}_g} \|c_g^{(m)} - \bar{c}_g\|_2^2. \end{aligned} \tag{5}$$

Here, $\mathcal{T}_g^{(m)}$ denotes the fit-side normal executions of exact task g produced by executor m, and $h_T(\tau)$ is the terminal process representation of trajectory τ. The mean representation $c_g^{(m)}$ summarizes task g under executor m, while $\bar{c}_g$ averages these source-specific means over the executor set $\mathcal{S}_g$. The alignment loss $\mathcal{L}_{\text{align}}$ pulls each $c_g^{(m)}$ toward $\bar{c}_g$, encouraging same-task executions to agree across executors while preserving differences between tasks.

A normal reference bank stores process-state and transition examples with task and event context:

$$\begin{aligned} \mathcal{B}_N &= \{(z_j, \Delta z_j, s_j)\}_{j \in \mathcal{D}_{\text{fit}}^N}, \\ \mathcal{N}_t &= \text{Retrieve}_K(q, a_{t+1}, \mathcal{B}_N), \\ w_j &= \frac{\exp(\beta\ \text{sim}_j)}{\sum_{\ell \in \mathcal{N}_t} \exp(\beta\ \text{sim}_\ell)}, \qquad \bar{z}_t^N = \sum_{j \in \mathcal{N}_t} w_j z_j. \end{aligned} \tag{6}$$

Here, $\mathcal{B}_N$ is the fit-side normal reference bank, where z_j is a normal process-state reference, Δz_j is its associated transition, and s_j summarizes the task and event context. The retrieval operator selects a local support set $\mathcal{N}_t$ compatible with the current task and candidate action. The weight w_j is the normalized similarity assigned to reference j, and $\bar{z}_t^N$ is the resulting weighted local normal center.

The compatibility score sim_j measures how well each reference matches the task semantics and candidate action/event context. A category-wide prototype is too restrictive when related tasks admit different valid process paths. Local retrieval preserves compatible normal alternatives without

forcing the category into a single center. This local support accommodates task-conditioned and potentially multimodal normal structure without requiring an explicit multimodal density model.

4.3 Predictive World Model

Given the process prefix, task, candidate action, and compatible normal context, the world model predicts the state reached after the action. Let $z_{t+1} = \Phi(o_{t+1})$ be the realized next process state, parameterized in the implementation by a structured outcome representation. The prediction is

$$\begin{aligned}(\hat{\mu}_{t+1}, \hat{\ell}_{t+1}) &= F_\phi(h_t, a_{t+1}, q, \mathcal{N}_t),\\ \hat{z}_{t+1} &= \hat{\mu}_{t+1}, \qquad o_{t+1} \notin \mathrm{input}(F_\phi).\end{aligned} \tag{7}$$

Here, h_t is the current process representation, a_{t+1} is the candidate next action, q denotes task semantics, and $\mathcal{N}_t$ is the retrieved local normal support. The world model F_ϕ outputs the predicted mean $\hat{\mu}_{t+1}$ and log variance $\hat{\ell}_{t+1}$ of the next process state, with $\hat{z}_{t+1} = \hat{\mu}_{t+1}$ used as the forecast state. The realized outcome o_{t+1} defines the training target $z_{t+1} = \Phi(o_{t+1})$ after execution and is never used as an input to its own prediction.

We learn normal one-step process dynamics with a Gaussian prediction objective:

$$\begin{aligned}\mathcal{L}_{\mathrm{pred}} &= \mathbb{E}_{\mathcal{D}^N_{\mathrm{fit}}}\left[\frac{1}{2}\sum_d \left((z_{t+1,d} - \hat{\mu}_{t+1,d})^2 e^{-\hat{\ell}_{t+1,d}} + \hat{\ell}_{t+1,d}\right)\right],\\ \mathcal{L}_{\mathrm{WM}} &= \mathcal{L}_{\mathrm{pred}} + \lambda_{\mathrm{align}}\mathcal{L}_{\mathrm{align}}.\end{aligned} \tag{8}$$

Here, $\mathcal{D}^N_{\mathrm{fit}}$ denotes the fit-side normal trajectories, and $\mathcal{L}_{\mathrm{pred}}$ is the Gaussian negative log-likelihood for next-state prediction. The coefficient λ_{align} controls the exact-task alignment term $\mathcal{L}_{\mathrm{align}}$, giving the full world model objective $\mathcal{L}_{\mathrm{WM}}$. This trains the model to capture normal one-step process dynamics while encouraging same-task consistency across executors.

4.4 Prospective Risk Fusion

A prefix can appear normal while a candidate action can move the process outside its normal support. We therefore score the predicted future, not only the current state. We first measure how far the predicted next state departs from its local normal support:

$$\begin{aligned}d^{\mathrm{raw}}_{t+1} &= \|\hat{z}_{t+1} - \bar{z}^N_t\|_2^2, \qquad & \rho_j &= \|z_j - \bar{z}^N_t\|_2^2,\\ s_t &= \max\{1.4826\ \mathrm{MAD}(\{\rho_j\}), 0.1 Q_{0.75}(\{\rho_j\}), \epsilon\},\\ d^{\mathrm{norm}}_{t+1} &= \frac{d^{\mathrm{raw}}_{t+1} - \mathrm{median}(\{\rho_j\})}{s_t}.\end{aligned} \tag{9}$$

Here, $\hat{z}_{t+1}$ is the predicted next process state and $\bar{z}^N_t$ is the weighted center of its local normal support. The quantity ρ_j measures the distance of reference z_j from this center, while s_t is a robust local scale computed from the median absolute deviation and upper-quartile spread of these reference distances. Thus, d^{raw}_{t+1} measures absolute future-state deviation and d^{norm}_{t+1} expresses the same deviation relative to the local normal neighborhood.

We further characterize the predicted future with four complementary signals:

$$\begin{aligned}E^{\mathrm{local}}_{t+1} &= \mathrm{median}_{j\in\mathcal{N}_t}\ \|\hat{z}_{t+1} - z_j\|_2^2,\\ d^{\Delta}_{t+1} &= \mathrm{median}_{\Delta z_j \in \Delta\mathcal{N}_t}\ \|(\hat{z}_{t+1} - z_t) - \Delta z_j\|_2^2,\\ d^{\mathrm{disp}}_{t+1} &= \mathrm{median}_{j\in\mathcal{N}_t}\ \|z_j - \bar{z}^N_t\|_2^2,\\ u_{t+1} &= D^{-1}\sum_{d=1}^{D}\exp(\hat{\ell}_{t+1,d}).\end{aligned} \tag{10}$$

Here, E^{local}_{t+1} measures support for the predicted state across normal references. The transition score d^{Δ}_{t+1} compares the predicted change $\hat{z}_{t+1} - z_t$ with compatible normal transitions in $\Delta\mathcal{N}_t$, where $z_t = \Phi(o_t)$ is the latest observed process state. The dispersion d^{disp}_{t+1} describes the spread of the

retrieved support, while u_{t+1} averages the predictive variance derived from the D-dimensional log-variance vector $\hat{\ell}_{t+1}$. No single statistic is assumed to define abnormality by itself.

The risk model uses four evidence views. The direct view x^{dir} contains task semantics and candidate-action structural, semantic, and lexical evidence available before execution. The dynamic view x^{dyn} contains completed process context, z_t, $\hat{z}_{t+1}$, and their predicted transition. The local view x^{loc} contains normal state and transition support and residuals to that support. The scalar view x^{sca} contains robustly scaled deviation, dispersion, and uncertainty statistics above. Thus the latter three views provide predicted-process evidence while the direct view supplies a legal baseline.

Let the view encoders produce same-width representations $d = G_d(x^{\mathrm{dir}})$, $y = G_y(x^{\mathrm{dyn}})$, $l = G_l(x^{\mathrm{loc}})$, and $s = G_s(x^{\mathrm{sca}})$. The final logit and score are

$$\begin{aligned} g &= \sigma(W_g[d; y; l; s] + b_g)\,, \\ f &= g \odot y + (1 - g) \odot d, \\ \eta_{t+1} &= \ell^{\mathrm{dir}}_{t+1} + H_\psi([f; l; s]), \\ r_{t+1} &= \sigma(\eta_{t+1}). \end{aligned} \tag{11}$$

Here, d, y, l, and s are the encoded direct, dynamic, local, and scalar evidence views, respectively. The gate g adaptively balances direct evidence d and predicted dynamics y, producing the fused representation f. The direct logit $\ell^{\mathrm{dir}}_{t+1}$ provides the pre-execution baseline, while H_ψ uses the fused, local, and scalar evidence to produce a predictive residual. Their sum gives the final logit η_{t+1}, which is mapped to the risk score r_{t+1} by a sigmoid.

We train the risk head with onset-aware supervision. For an anomalous trajectory with first onset k,

$$\begin{aligned} \mathcal{D}_{\mathrm{risk}}(\tau) &= \{(x_i, 0) : i < k\} \cup \{(x_k, 1)\}, \\ \mathcal{L}_{\mathrm{risk}} &= \mathcal{L}_{\mathrm{WBCE}} + \lambda_r \max\left(0,\ m - \eta_k + \max_{i<k} \eta_i\right). \end{aligned} \tag{12}$$

Here, x_i collects the four evidence views for candidate i, and $\mathcal{D}_{\mathrm{risk}}(\tau)$ contains all pre-onset negatives together with the onset positive. The weighted binary cross-entropy $\mathcal{L}_{\mathrm{WBCE}}$ provides step-level supervision, while the ranking term with weight λ_r and margin m encourages the onset logit η_k to exceed the largest pre-onset logit $\max_{i<k} \eta_i$. Excluding post-onset steps keeps supervision focused on the first transition into abnormal execution. The predictive model and normal reference bank are learned or constructed from fit-side normal trajectories, after which the risk head is trained with fit-side onset supervision. At inference, the score is computed from only the completed prefix, task context, candidate action, and fit-normal references before execution.

5 Experiments

5.1 Experimental Setup

Data and prospective boundary. We study 23,857 trajectories from 25 raw subsets: 20,322 normal and 3,535 anomalous trajectories with a valid first onset. An exact task is the pair of task ID and normalized question. For an anomalous trajectory, steps before the first onset are negative, the onset step is the sole positive, and subsequent steps are excluded; eligible normal steps are negative. At decision time, the auditor sees the task/question, completed action–result prefix, and pending action. It cannot use the pending result, future steps or trajectory length, source/subset identity, or labels. Retrieval uses fit-side normal trajectories only.

Splits and selection. Six held-source, five held-exact-task, and 25 strict leave-one-raw-subset-out (LOSO) splits at seeds 2026, 2027, and 2028 yield 108 split–seed runs. The Q1/Q2 representation and world-model route are held fixed across Q3 comparisons. The task-disjoint confirmation protocol partitions each original validation split by exact task into approximately 70% development and 30% confirmation. Development alone selects the risk epoch and thresholds from normal trajectory maximum scores; confirmation tasks are disjoint from development tasks within validation. The primary operating point targets 10% development normal trajectory FAR, with 5% and 20% auxiliary points. We report First-Alarm Exact-Onset Hit, requiring the *first* alarm at onset, together with observed normal Trajectory FAR.

Table 1: One-time external silver world-model evaluation, separate from task-disjoint confirmation. All routes predict the same 29-dimensional next structural outcome. Each route cell reports normal next-state MSE (top) and onset-truncated predictive-deviation AP (bottom); lower MSE is better. Paired retrieval-minus-comparator MSE 95% confidence intervals are reported in 10^{-3} units.

Regime	Task-only	MoE	Retrieval	Ret.–Task CI	Ret.–MoE CI
	0.0504	0.0494	**0.0487**		
Held source	0.1035	0.1010	0.1070	$[-2.073, -1.019]$	$[-1.271, +0.003]$
	0.0480	0.0476	**0.0471**		
Held task	0.0921	0.0875	0.0979	$[-1.071, -0.743]$	$[-0.738, -0.334]$
	0.0460	0.0457	**0.0451**		
LOSO	**0.1407**	0.1367	0.1370	$[-1.176, -0.586]$	$[-0.852, -0.202]$

Comparators. Matched onset-supervised heads are R-A (direct task/action), R-W (world-model-side views), R-F (concatenated views), and R-G (gated fusion). E2 Automata and E3 Sparse-Evidence are causal adaptations trained with trajectory labels. TS-Traj and TS-Onset adapt the ToolSafe/TS-Guard architecture with a fixed observer; TS-Traj uses trajectory labels and TS-Onset uses onset labels. They are architecture adaptations rather than reproductions on ToolSafe's original unsafe-tool task. All eight Q3 methods share the split, development/confirmation, and FAR protocol; their supervision and model scale remain distinct. Detailed configurations and diagnostic analyses appear in Appendix B.

5.2 Normal Structure

In the nine-run development geometry study, validation exact-task cross-source distance falls from 0.1286 at $\lambda = 0$ to 0.0882 at $\lambda = 0.10$ (31.4% contraction), while different-subset distance also falls (0.1941 to 0.1154) and source-ID 1NN accuracy does not decrease (0.4390 to 0.4505). This supports cross-source contraction, not source invariance (Figure 4; Table 5). Local task/event support improves the development comparison over an event centroid, while the separate held-subset LOSO result is not consistently favorable; we therefore use it as a conditional reference rather than evidence for a universal compact manifold (Table 6).

5.3 World Modeling

Table 1 compares routes that predict the same 29-dimensional next structural outcome. Retrieval has the lowest mean normal MSE on held source, held exact task, and LOSO in the one-time external silver evaluation. Its LOSO MSE is lower than task-only in 22/25 seed-averaged subsets and MoE in 17/25. The advantage is small; the cross-model retrieval-minus-MoE MSE interval touches zero.

On LOSO, retrieval predictive-deviation AP is 0.1370 versus task-only's 0.1407, so lower MSE does not necessarily improve anomaly discrimination. Action-conditioning controls support nontrivial action-conditioned prediction; full control results are in Appendix B.3.

5.4 Onset Risk

At the primary 10% development Normal-trajectory FAR budget, R-G achieves a 56.8% exact first-alarm hit at 10.3% observed FAR, compared with 39.5% at 10.1% FAR for direct-only R-A (Table 2). This is a 17.3 percentage-point gain (95% CI [16.2, 18.4] pp), while AP increases from 35.0% to 51.7% and AUROC from 93.0% to 95.3%. The progression from R-A to R-W (50.4%), R-F (54.8%), and R-G (56.8%) shows that predicted process views provide information beyond direct task/action compatibility. R-G also exceeds the trajectory-supervised Automata and Sparse-Evidence adaptations, and matches the trajectory-supervised TS-Traj adaptation in exact hit (56.8% vs. 56.8%). TS-Onset attains the highest absolute hit at 62.4%, exceeding R-G by 5.62 pp, but uses a frozen 7.07B-parameter observer, whereas R-G loads 123K parameters. We therefore treat this as a performance–cost trade-off rather than a universal accuracy advantage; Section 5.5 analyzes the efficiency difference. The TS variants are architecture adaptations to our task rather than reproductions of the original unsafe-tool setting. Figure 3 shows the development-calibrated FAR–hit frontier.

5.5 Efficiency and Generalization

We examine runtime efficiency. R-G loads 122,940 parameters, has 7.07 ms batch-1 p50 latency, and reaches 56.8% hit. TS-Traj loads 7.071B parameters and takes 212.25 ms for the same 56.8%

Table 2: Task-disjoint confirmation under a shared split, development-selection, and FAR-calibration protocol (108 split–seed runs; primary 10% development normal-trajectory FAR budget). Hit is exact first-alarm onset hit. Loaded parameters are rounded and include fixed components. Latency values are batch-1 p50 measurements; TS detection scores and profiling were obtained in separate BF16 and float32 runs. E2/E3 were not profiled on the same GPU. TS rows are ToolSafe/TS-Guard architecture adaptations.

Method	Supervision	Params	p50 (ms)	Hit↑ (%)	FAR↓ (%)	AP↑ (%)	AUROC↑ (%)
R-A	onset	20K	2.40	39.5	10.1	35.0	93.0
R-W	onset	100K	6.90	50.4	10.2	42.6	94.3
R-F	onset	120K	6.95	54.8	10.2	49.9	95.2
R-G	onset	**123K**	**7.07**	**56.8**	10.3	51.7	95.3
E2 Automata	trajectory	–	–	36.0	9.9	15.9	81.5
E3 Sparse-Evidence	trajectory	–	–	20.8	10.1	17.1	77.8
TS-Traj	trajectory	7.07B	212.3	56.8	10.1	52.5	93.0
TS-Onset	onset	**7.07B**	**212.9**	**62.4**	10.0	**58.3**	**96.3**

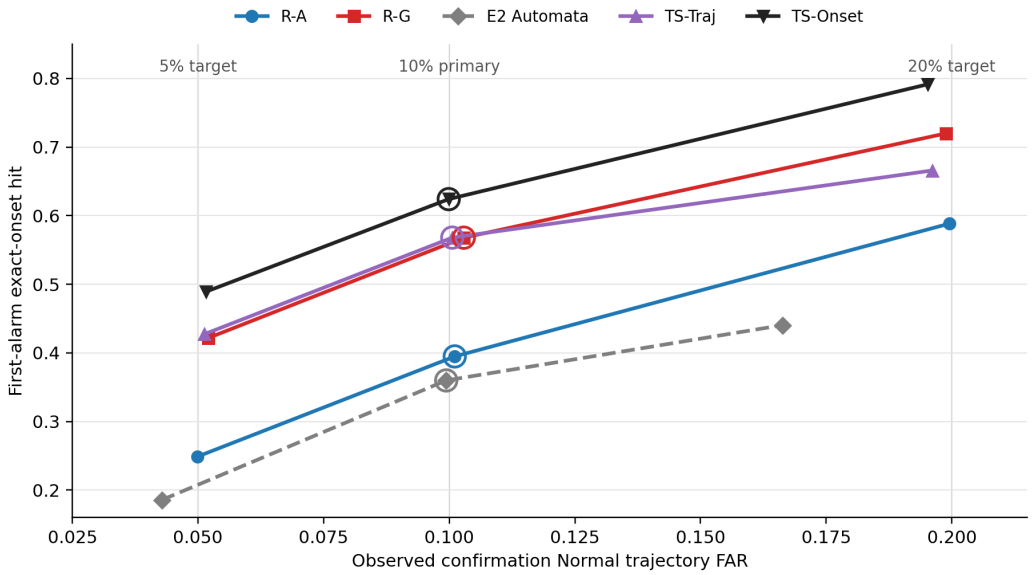


Figure 3: Development-calibrated FAR–hit frontier for five representative methods under the shared task-disjoint confirmation protocol. Thresholds use development normal trajectories only; circled points mark the 10% primary budget, with 5% and 20% auxiliary guides. Exact hit can be non-monotonic when earlier alarms invalidate onset localization.

hit, corresponding to 30.0× R-G's p50 latency. TS-Onset reaches 62.4% hit, 5.62 pp higher than R-G, with 212.9 ms p50 latency (30.1×) and 57,520× loaded parameters. The profiler uses one RTX 4090, float32, and development prefixes; TS detection scores were obtained separately with a BF16 observer. These measurements characterize inference and model scale rather than FLOPs or total training cost. The parameter comparison and p95 latency are reported in Figure 8 and Table 12.

Across the task-disjoint confirmation transfer regimes, R-G exceeds R-A and modestly exceeds R-F in hit (Appendix Table 8). TS-Onset is stronger on cross-model and LOSO; the held-exact-task R-G-minus-TS-Onset paired CI [-7.88,+1.85] pp crosses zero. Held-subset results show heterogeneous multi-turn, SQL, and JavaScript behavior; per-subset and FAR-tail results are in Appendix A.

6 CONCLUSION

We study prospective process risk in LLM agents by predicting how a candidate action changes execution. Normal behavior is better modeled with task-conditioned local support than a category-wide center, motivating retrieval-conditioned next-state prediction. Combining task-action evidence with predicted state and transition deviations improves first-alarm exact-onset detection by 17.3 pp over the matched direct-only baseline at similar 10% normal-trajectory FAR, supporting predictive process modeling as an auxiliary signal for agent auditing.

AI USE STATEMENT

Generative AI tools were used solely to aid and polish the writing of this paper, including improving grammar, clarity, and readability. All AI-assisted edits were reviewed by the authors. The authors take full responsibility for the final content of the paper.

ETHICS STATEMENT

This work studies process-risk auditing for LLM agents with the goal of identifying potentially unreliable execution before a candidate action is carried out. The experiments use benchmark-derived agent trajectories and do not intentionally collect new human-subject data. The proposed risk score may produce both false alarms and missed detections, especially under distribution shift, and should therefore be regarded as an auxiliary auditing signal rather than a standalone guarantee of agent safety. We do not use the method to enable harmful agent behavior or expand agents' operational capabilities.

REPRODUCIBILITY STATEMENT

We report the dataset composition, prospective prediction boundary, split protocols, random seeds, checkpoint-selection rules, and runtime threshold-calibration procedure in the main paper. Additional model configurations, training hyperparameters, and implementation details are provided in the supplementary material. All reported comparisons follow the stated held-source, held-exact-task, and leave-one-subset-out protocols, with seeds 2026, 2027, and 2028 where applicable. Full per-subset results and additional experimental diagnostics are also included in the supplementary material.

REFERENCES

Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. *Advances in neural information processing systems*, 37:82895–82920, 2024.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

Jiaqian Li, Yanshu Li, Boxuan Zhang, Ruixiang Tang, and Kuan-Hao Huang. Traces: Proactive safety auditing for multi-turn llm agents via trajectory-state modeling. *arXiv preprint arXiv:2605.27690*, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Yibing Liu, Yangze Liu, Xiaolong Yin, Bin Wang, Chong Zhang, Hao Yin, and Zhongyi Han. Openclawbench: Benchmarking process-side anomalies in real-world agent execution trajectories. *arXiv preprint arXiv:2605.29253*, 2026.

Yutao Mou, Zhangchi Xue, Lijun Li, Peiyang Liu, Shikun Zhang, Wei Ye, and Jing Shao. Toolsafe: Enhancing tool invocation safety of llm-based agents via proactive step-level guardrail and feedback. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 37125–37153, 2026.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris Maddison, and Tatsunori Hashimoto. Identifying the risks of lm agents with an lm-emulated sandbox. In *International Conference on Learning Representations*, volume 2024, pp. 27031–27098, 2024.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020. URL https://EconPapers.repec.org/RePEc:nat:nature:v:588:y:2020:i:7839:d:10.1038_s41586-020-03051-4.

Hao Tang, Darren Key, and Kevin Ellis. Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment. *Advances in Neural Information Processing Systems*, 37:70148–70212, 2024.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Hongfei Xia, Hongru Wang, Zeming Liu, Qian Yu, Yuhang Guo, and Haifeng Wang. SafeToolBench: Pioneering a prospective benchmark to evaluating tool utilization safety in LLMs. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 17643–17660, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.958. URL https://aclanthology.org/2025.findings-emnlp.958/.

Yuejin Xie, Youliang Yuan, Wenxuan Wang, Fan Mo, Jianmin Guo, and Pinjia He. Toolsafety: A comprehensive dataset for enhancing safety in llm-based agent tool invocations. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 14146–14167, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Fangqi Li, Zhuosheng Zhang, et al. R-judge: Benchmarking safety risk awareness for llm agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1467–1490, 2024.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 10471–10506, 2024.

Zhexin Zhang, Shiyao Cui, Yida Lu, Jingzhuo Zhou, Junxiao Yang, Hongning Wang, and Minlie Huang. Agent-safetybench: Evaluating the safety of llm agents. *arXiv preprint arXiv:2412.14470*, 2024.

Siyu Zhou, Tianyi Zhou, Yijun Yang, Guodong Long, Deheng Ye, Jing Jiang, and Chengqi Zhang. Wall-e: World alignment by neurosymbolic learning improves world model-based llm agents. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 65352–65391. Curran Associates, Inc., 2025. doi: 10.52202/085713-2192. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/5e772a13ccba5255331240dcd99aa38b-Paper-Conference.pdf.

A Per-subset Leave-one-subset-out Results

The following held-subset LOSO results are descriptive and separate from the task-disjoint confirmation set. Each row averages three seeds for one held raw BFCL subset at a 10% development-normal trajectory FAR budget. Anomaly N is the onset-positive count for one split, not a three-seed sum. The last column is R-G minus R-A first-alarm hit. Small subsets have unstable rates.

Table 3: All 25 descriptive held raw-subset LOSO results. R-G/R-A/R-F hit is first-alarm exact-onset hit; R-G FAR is normal trajectory FAR; SD is sample seed standard deviation of R-G hit. Every value is descriptive and uses the original development threshold, with no subset-specific calibration.

Raw subset	Anomaly N	R-G hit	R-A hit	R-F hit	R-G FAR	R-G seed SD	Δ hit
BFCL_v3_chatable	109	.5719	.4190	.5749	.1028	.0191	+.1529
BFCL_v3_exec_multiple	29	.5287	.3333	.5172	.1034	.0398	+.1954
BFCL_v3_exec_parallel	34	.5490	.3627	.5098	.1022	.0340	+.1863
BFCL_v3_exec_parallel_multiple	35	.4190	.2000	.4095	.1530	.0719	+.2190
BFCL_v3_exec_simple	48	.6181	.2778	.6111	.0977	.0318	+.3403
BFCL_v3_irrelevance	196	.6854	.4473	.6956	.0721	.0511	+.2381
BFCL_v3_java	127	.5827	.5302	.5197	.1586	.1042	+.0525
BFCL_v3_javascript	52	.6026	.4038	.5769	.2197	.0444	+.1987
BFCL_v3_live_irrelevance	305	.4503	.3749	.4590	.0589	.0095	+.0754
BFCL_v3_live_multiple	878	.6503	.4339	.6469	.1299	.0254	+.2164
BFCL_v3_live_parallel	16	.5625	.1458	.3750	.1152	.2165	+.4167
BFCL_v3_live_parallel_multiple	12	.5833	.5833	.6111	.0947	.0833	+.0000
BFCL_v3_live_relevance	4	.5000	.2500	.4167	.1886	.0000	+.2500
BFCL_v3_live_simple	162	.4053	.2819	.3889	.0628	.0198	+.1235
BFCL_v3_multi_turn_base	103	.4272	.2654	.4790	.0909	.1027	+.1618
BFCL_v3_multi_turn_composite	79	.3418	.2911	.3376	.0750	.0335	+.0506
BFCL_v3_multi_turn_long_context	110	.4030	.2909	.3788	.1047	.0430	+.1121
BFCL_v3_multi_turn_miss_func	91	.4176	.2894	.4322	.1209	.0190	+.1282
BFCL_v3_multi_turn_miss_param	95	.3088	.1930	.3825	.0705	.0338	+.1158
BFCL_v3_multiple	179	.6760	.4227	.6555	.1152	.0436	+.2533
BFCL_v3_parallel	158	.6498	.5021	.6667	.1280	.0349	+.1477
BFCL_v3_parallel_multiple	194	.7423	.6048	.7302	.1904	.0458	+.1375
BFCL_v3_rest	87	.3908	.3257	.4100	.1763	.0414	+.0651
BFCL_v3_simple	338	.6874	.4783	.6568	.1243	.0239	+.2091
BFCL_v3_sql	94	.1418	.0993	.0922	.0212	.0491	+.0426

B Additional Experimental Protocol and Diagnostics

These supplementary results retain their original evidence units. Table 4 restates the protocol. Q1 is a nine-run development geometry study; Q2 comprises a one-time external silver comparison and development action-conditioning controls; Q3 tables distinguish task-disjoint confirmation from descriptive held-subset diagnostics.

B.1 Implementation Details

Process encoder and world model. The process encoder uses a GRU with hidden size 64. The fit-normal reference bank retrieves at most $K = 8$ task/event-compatible neighbors. The world model predicts a 29-dimensional next structural outcome. We set $\lambda_{\text{align}} = 0.1$ and train the world model on fit-normal trajectories with AdamW (learning rate 10^{-3}, weight decay 10^{-4}), batch size 128, for 8 epochs. The checkpoint minimizes validation-Normal next-outcome MSE.

Risk heads. Each of the direct, dynamic, local, and scalar views is encoded to 32 dimensions; R-G uses a 32-dimensional sigmoid gate. The four matched onset-supervised heads use full-batch AdamW (learning rate 3×10^{-3}, weight decay 10^{-4}) for 80 epochs. The weighted BCE positive weight is $\min(20, N_-/N_+)$. For trajectories with pre-onset steps, the ranking loss uses $\lambda_r = 0.5$ and margin $m = 0.5$. Candidate epochs 20, 40, 60, and 80 are selected by development first-alarm exact-onset hit at a 10% development-Normal trajectory FAR budget; ties favor lower pre-onset alarm rate, higher AP, then the earlier epoch. The threshold is calibrated on development-Normal trajectory maximum scores and applied unchanged to the task-disjoint confirmation partition.

Table 4: Experimental protocol. The confirmation set is an exact-task-disjoint fraction of each original validation partition, not an untouched external test set.

Item	Protocol
Development release	23,857 trajectories: 20,322 normal, 3,535 anomalous with valid first onset; 25 raw subsets.
Transfer splits	Six held source, five held exact task, 25 strict LOSO; seeds 2026/2027/2028, for 108 split–seed runs.
Prospective input	Completed prefix, task/question, pending action; no pending result, future, trajectory length, source/subset, or label as a forward feature.
Risk selection	Approximately 70/30 exact-task-disjoint development/confirmation within original validation; development-only epoch and normal-trajectory-maximum threshold selection.
Primary endpoint	First-Alarm Exact-Onset Hit and observed normal Trajectory FAR at a 10% development normal FAR budget; 5%/20% auxiliary.

B.2 Protocol and Q1 structure

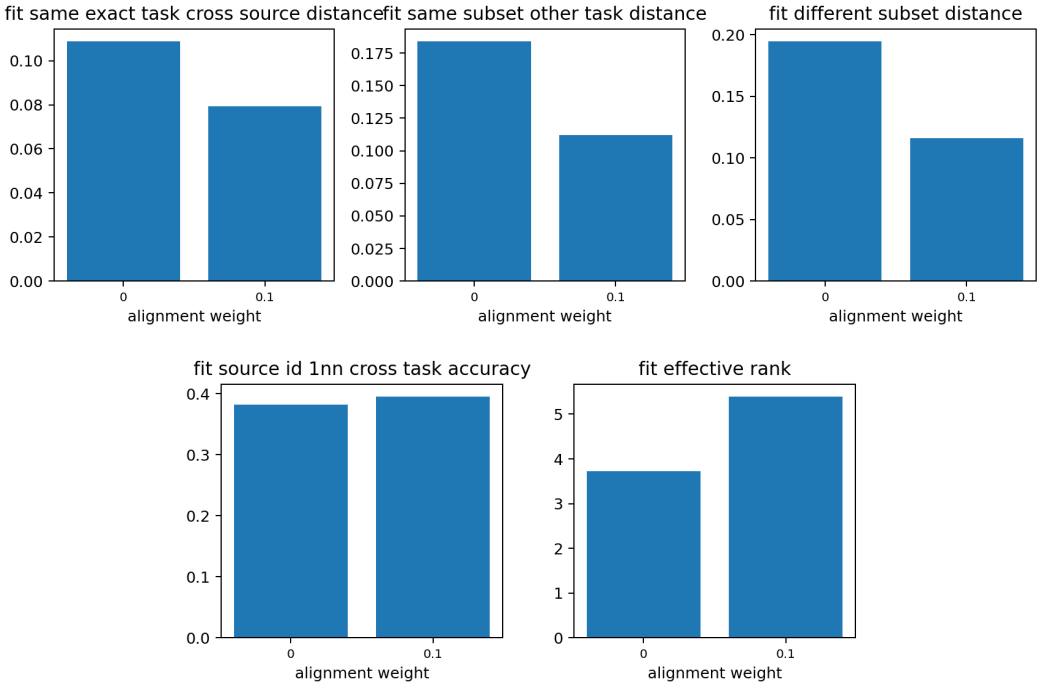


Figure 4: Descriptive development geometry study of fit-normal terminal-state structure (nine split–seed runs per weight). Validation exact-task cross-source distance contracts at $\lambda = 0.10$, but different-subset distance also contracts and source-ID 1NN rises; absolute contraction does not establish source invariance. This geometry study was not used to select the final predictive risk model.

Table 5: Validation normal terminal-state geometry in the nine-run development geometry study. Distances, source-ID 1NN accuracy, effective rank, and local eight-neighbor dispersion are descriptive; all weights are shown.

λ	Exact-task cross-source	Same-subset other-task	Different-subset	Source-ID 1NN	Rank / local dispersion
0	.12858	.19033	.19405	.43902	3.75729 / .03290
.05	.09925	.13246	.13553	.44379	4.84375 / .02990
.10 (fixed)	.08819	.11235	.11541	.45052	5.47282 / .02846
.20	.07464	.09292	.09558	.45399	6.28770 / .02641

Table 6: Earlier scalar-risk local-versus-centroid comparison at the 10% budget. Validation favors local support; the separate held-subset LOSO evaluation reverses numerically, with a paired interval crossing zero. The intervention retrained the world model and head.

Evidence	Local hit	Event-centroid hit
Validation equal-regime macro	.25938	.23380
Separate held-subset LOSO	.16879	.17784

In the separate validation comparison with the earlier scalar-risk formulation, local task/event hit is 0.25938 versus 0.23380 for the event centroid (Table 6). The held-subset LOSO result reverses numerically (0.16879 versus 0.17784), with a paired interval crossing zero. Because this intervention retrained the world model and head, it does not isolate a causal contribution to final R-G risk.

B.3 Q2 CONTROLS

Table 7: Nine-run development action-conditioning analysis of normal next-structural-outcome MSE. The shuffled condition perturbs only the direct action tensor in about 50.07% of eligible rows; candidate semantics and retrieval remain. No-action fitting zeros direct candidate action and candidate reference.

Condition	normal MSE
Full retrieval route	.045238
Shuffled direct candidate action	.045807
No-action fit	.051745
Copy preceding structural outcome	.198749

B.4 Q3 SUPPLEMENTARY COMPARISONS

Table 8 gives task-disjoint confirmation results by transfer regime. Table 9 gives pre-onset rates and paired hit intervals. Table 10 and Figure 5 retain all eight methods and six calibration budgets. Figure 6 shows conditional onset-delay distributions; Table 11 reports unconditional timing rates.

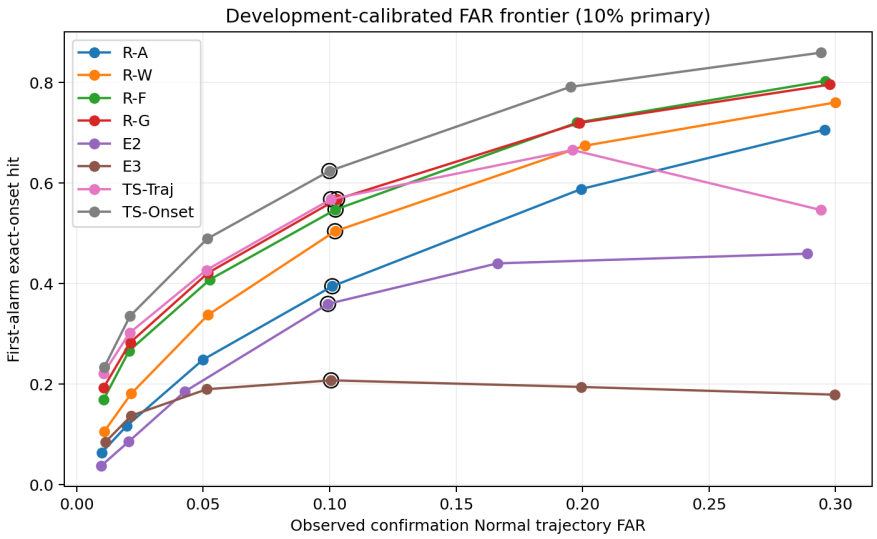


Figure 5: Full eight-method development-calibrated confirmation FAR–hit frontier over six budgets. Circled markers indicate the 10% primary operating point; the remaining budgets are descriptive.

Table 8: Task-disjoint confirmation by transfer regime at the primary budget. Each cell reports first-alarm exact-onset Hit (%) / observed normal trajectory FAR (%), averaged over split–seed runs within each regime. Separate held-subset tails are excluded.

Method	Cross-model (6×3)	Held exact task (5×3)	LOSO (25×3)
R-A	35.9 / 9.7	46.9 / 10.3	38.9 / 10.2
R-F	50.4 / 10.5	60.6 / 9.8	54.6 / 10.3
R-G	**53.5** / 10.6	**61.9** / 9.9	**56.6** / 10.3
TS-Traj	53.7 / 10.3	59.2 / 9.7	57.1 / 10.1
TS-Onset	**59.3** / 10.6	**65.3** / 9.8	**62.6** / 9.9

Table 9: Additional matched confirmation diagnostics at the primary 10% budget. Differences are R-G minus the comparator in exact first-onset hit; intervals resample 36 held-split means after averaging three seeds. Pre-onset alarm is a step rate, not normal trajectory FAR.

Comparator	Pre-onset step alarm	R-G minus comparator hit, 95% CI
R-A	.0022	+.1730 [.1620,.1844]
R-W	.0042	+.0636 [.0534,.0738]
R-F	.0040	+.0204 [.0146,.0263]
R-G	.0049	reference
E2 Automata	.0474	+.2080 [.1903,.2262]
E3 Sparse-Evidence	.2450	+.3601 [.3404,.3797]
TS-Traj	.0017	-.0001 [-.0133,.0134]
TS-Onset	.0033	-.0562 [-.0684,-.0434]

Table 10: Full matched confirmation FAR–hit frontier. Each cell is observed normal trajectory FAR / exact first-alarm onset hit; column headings are development normal FAR calibration budgets. The 10% column is primary.

Method	1%	2%	5%	10%	20%	30%
R-A	.0098/.0638	.0197/.1167	.0499/.2489	.1011/.3949	.1995/.5883	.2957/.7064
R-W	.0109/.1065	.0215/.1814	.0519/.3379	.1021/.5043	.2009/.6744	.2999/.7604
R-F	.0106/.1689	.0207/.2665	.0527/.4084	.1023/.5475	.1978/.7204	.2959/.8032
R-G	.0106/.1933	.0212/.2829	.0520/.4216	.1029/.5679	.1988/.7198	.2977/.7959
E2	.0097/.0378	.0205/.0857	.0428/.1857	.0994/.3599	.1663/.4404	.2888/.4596
E3	.0114/.0856	.0216/.1378	.0515/.1902	.1006/.2078	.1993/.1947	.2996/.1794
TS-Traj	.0106/.2220	.0210/.3023	.0512/.4271	.1006/.5680	.1961/.6658	.2942/.5465
TS-Onset	.0109/.2344	.0211/.3358	.0516/.4892	.0999/.6241	.1952/.7914	.2943/.8594

Table 11: First-alarm timing at the fixed 10% operating point, with all anomalous trajectories as denominator. Preventive windows count onset or earlier alarms within w steps; retrospective $|\delta|$ windows also admit late alarms and cannot be read as prospective success.

Method	Exact	Preventive $w = 1$	$w = 2$	$w = 3$	$\lvert\delta\rvert \le 1$	$\lvert\delta\rvert \le 2$
R-G	.5679	.5710	.5749	.5788	.5816	.5900
R-A	.3949	.3952	.3956	.3972	.4051	.4082
E2	.3599	.3621	.3664	.3723	.3929	.4050
TS-Traj	.5680	.5696	.5705	.5706	.5984	.6120
TS-Onset	.6241	.6274	.6314	.6337	.6429	.6526

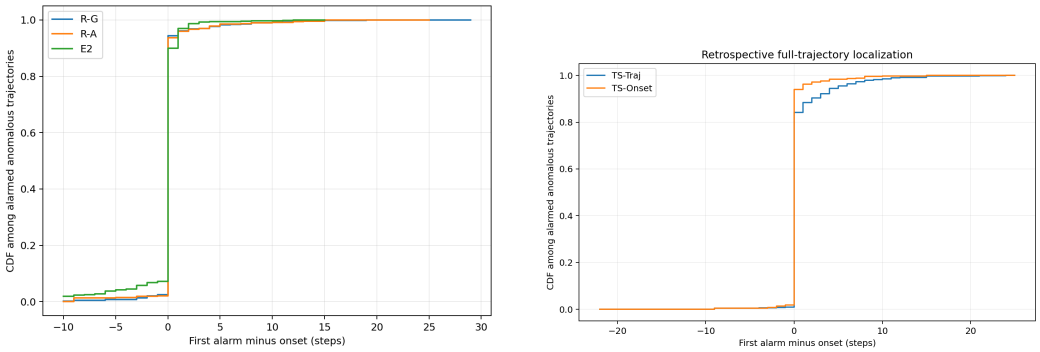


Figure 6: Onset-delay distributions for R-A, R-G, and E2 (left) and TS-Traj and TS-Onset (right) from separate evaluations. These CDFs condition on a first alarm; coverage differs by method and should be interpreted together with Table 11. Positive delay denotes a late alarm in retrospective full-trajectory replay.

B.5 Efficiency and label-budget diagnostics

Table 12 reports the measured p95 latency tail. Figure 7 shows the hit–latency comparison, and Figure 8 shows parameter scale. Table 13 and Figure 9 report the separate fixed-epoch label-budget study.

Table 12: Same-GPU float32 batch-1 profiling. End-to-end latency includes raw prefix processing but excludes checkpoint loading; p95 shows a long tail. Hit values use the fixed 10% task-disjoint confirmation point, with BF16 TS observers. Peak allocated GPU memory is not total device VRAM.

Method	p50 / p95 ms	Loaded parameters	Peak allocated
R-G	7.07 / 33.17	122,940	11.18 MiB
TS-Traj	212.25 / 1939.60	7,071,544,321	31.19 GiB
TS-Onset	212.92 / 1942.78	7,071,544,321	31.19 GiB

Table 13: Secondary fixed-epoch onset-label study: equal-regime macro confirmation hit. Normal fit data remain available, while onset labels are sampled by exact task. These fixed-epoch values have a different selection and aggregation protocol from the main 108-run table.

Onset labels	R-A	R-G	TS-Onset	R-G–R-A (95% CI)	R-G–TS-Onset (95% CI)
10%	.3175	.4709	.5256	+.1534 [.1427,.1647]	-.0547 [-.0705,-.0399]
25%	.3706	.5273	.5592	+.1567 [.1434,.1699]	-.0319 [-.0506,-.0161]
50%	.3968	.5521	.5935	+.1553 [.1398,.1704]	-.0415 [-.0602,-.0242]
100%	.4027	.5715	.6197	+.1688 [.1532,.1840]	-.0481 [-.0652,-.0307]

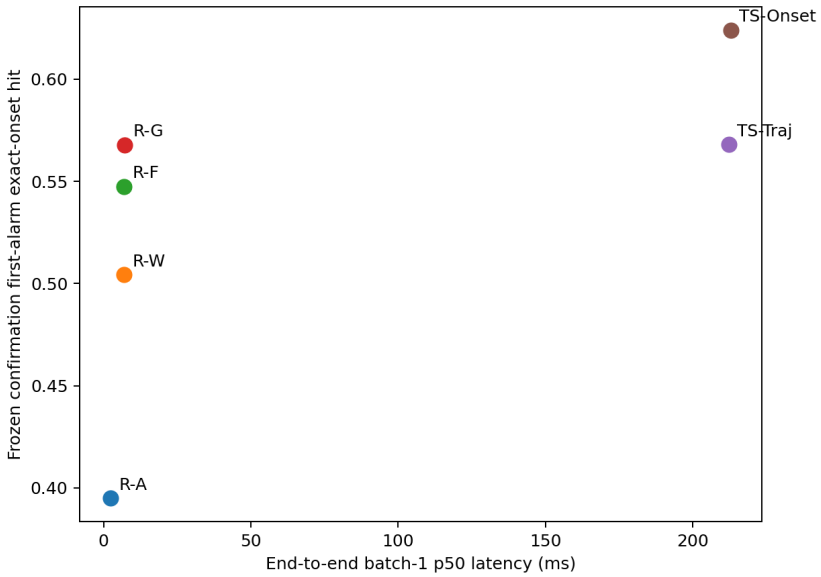


Figure 7: Exact hit at the fixed 10% operating point versus batch-1 p50 end-to-end latency on the same GPU. The float32 profiling and BF16 TS-Guard detection evaluations are separate; checkpoint loading is excluded. E2/E3 lack a comparable GPU profile.

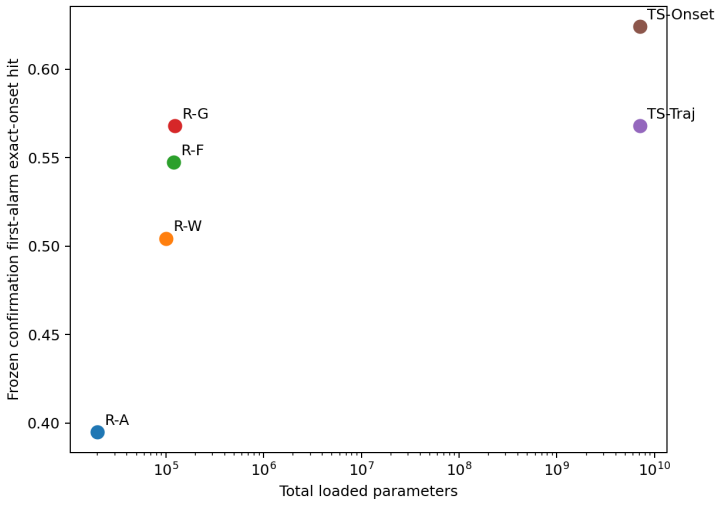


Figure 8: Loaded parameters versus exact hit at the fixed 10% operating point. E2/E3 lack a comparable parameter profile. The 7.071B TS counts include the fixed observer and head.

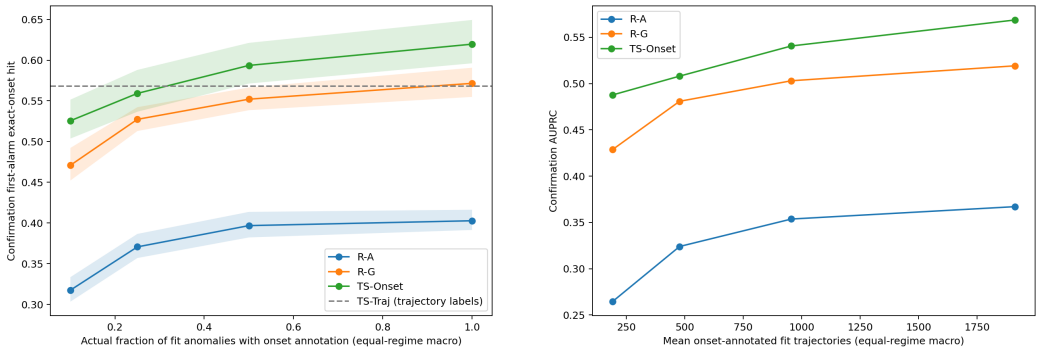


Figure 9: Secondary fixed-epoch label-budget analyses. Left: confirmation exact-onset hit versus the fraction of available onset annotations. Right: confirmation exact-onset hit versus the mean number of onset-labeled trajectories. The 100% points use equal-regime macro aggregation and should not be substituted for the development-selected main results.

B.6 Supplementary ablations, subset tails, and cache audit

Table 14: Separate diagnostic interventions not evaluated under the full 108-run matched task-disjoint confirmation protocol. These results are reported only to characterize specific components and should not be interpreted as full-grid feature-removal ablations.

Intervention family	Supported reading
Fixed held-task transition shuffle / local-view removal	Nominal hit changes alongside FAR; no isolated full-grid R-G feature necessity follows.
Fixed held-task completed-history removal	History removal changes R-G hit .5528 to .4720; equal-actual-FAR hit .5528 to .4845.
Full R-G single-feature removal	Not measured under the 108-run matched task-disjoint confirmation protocol.

The separate 25-subset LOSO table (Table 3) is descriptive rather than confirmation. Its 25 subset-mean R-G hit/FAR values average .5158/.1151; subset-mean FAR p95 is .19005. Across 75 subset–seed FAR rows, p95 is .197965 and the maximum is .262570 (JavaScript, seed 44). These units differ from confirmation LOSO R-G FAR p95 .12672. Multi-turn, SQL, and JavaScript subsets show heterogeneous performance, and low-count subsets do not support stable category claims.

The corrected normal-memory cache key includes task, effective action event, excluded task, retrieval count, and semantic tensor content. The earlier coarse key mismatched support in 174/452 sampled queries. Reversed, permuted, warm, and production batch/order variants passed under the corrected key, with unchanged fit-normal bank membership; the audit found no label leakage.