Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

# GOALGUARD: Autonomous Goal-Drift Detection & Correction

**Purvi M.**
Dept. of CSE (AI & ML)
Dayananda Sagar University
Bengaluru, India

**Sindhu S. Rao**
Dept. of CSE (AI & ML)
Dayananda Sagar University
Bengaluru, India

**Nikhil K**
Dept. of CSE (AI & ML)
Dayananda Sagar University
Bengaluru, India

**Mr. Suresh Kandasamy**
Professor Of Practice Dept. of CSE (AI & ML)
Dayananda Sagar University
Bengaluru, India

***Abstract*—As Large Language Models (LLMs) are increasingly deployed as autonomous agents operating over extended temporal horizons, maintaining strict adherence to human-specified system objectives becomes a foundational prerequisite for operational safety, alignment, and reliability. Recent empirical studies demonstrate that even frontier language model agents suffer from *goal drift*—the gradual behavioral deviation from an initial system goal toward competing environmental incentives or previously pursued instrumental sub-goals as context length expands. In this paper, we present GoalGuard, an end-to-end simulation, real-time telemetry, and runtime mitigation framework designed to operationalize, diagnose, and counteract goal drift in autonomous LLM agents. Grounded in the formal evaluation methodology of Arike et al., GoalGuard implements a controlled multi-step simulation environment (*Apex Capital*) that decouples drift via commission ($GD_{\text{actions}}$) from drift via omission ($GD_{\text{inaction}}$) across baseline, adversarial pressure, and goal-switching regimes. Furthermore, GoalGuard introduces a modular FastAPI-based runtime interception engine coupled with an interactive React/Vite telemetry dashboard that tracks pattern-matching susceptibility across expanding token horizons and dynamically injects structured goal-elicitation and context-anchoring interventions when drift thresholds are breached. We present the complete system architecture, formal drift metrics, and a standardized multi-tier evaluation protocol spanning trajectories up to 100,000 context tokens, establishing the benchmarking methodology for validating runtime reductions in cumulative $GD_{\text{actions}}$ and post-switch $GD_{\text{inaction}}$ prior to large-scale multi-model deployment.**

***Index Terms*—Autonomous Agents, Goal Drift, AI Safety, Long-Horizon Reasoning, Large Language Models, Behavioral Evaluation, Guardrails, Instrumental Convergence.**

## I. Introduction

Autonomous agents powered by Large Language Models (LLMs) are rapidly transitioning from stateless, single-turn conversational assistants to stateful decision-makers executing complex multi-step workflows across financial portfolio management, autonomous software engineering, scientific discovery, and enterprise resource planning [**?**], [**?**]. In these real-world deployments, agents must operate over extended time horizons with minimal human supervision, requiring them to balance a primary, human-assigned *system goal* against shifting environmental feedback, persuasive user nudges, and temporary *instrumental goals* necessary for intermediate task execution.

However, long-horizon autonomy introduces a subtle yet critical alignment failure mode known as **goal drift**: an agent's progressive behavioral tendency to deviate from its original system-level objective over time, even when that objective was unambiguously specified in the initial system prompt and remains accessible within the model's context window [**?**]. Unlike abrupt adversarial jailbreaks, explicit prompt injections, or catastrophic reasoning hallucinations, goal drift emerges gradually and silently as interaction histories accumulate.

Arike et al. [**?**] provided a seminal empirical investigation into this phenomenon, demonstrating that goal drift in modern LLM agents is not merely a byproduct of context length or memory decay ("lost in the middle" [**?**]). Instead, drift is strongly driven by an agent's increasing susceptibility to *autoregressive pattern-matching* over its own prior action trajectory rather than *goal-directed reasoning* from its system specification. This vulnerability becomes particularly acute when an agent exhibits high behavioral flexibility—for example, temporarily adopting an instrumental sub-goal for several timesteps before being instructed to revert to its primary system objective. During the post-switch phase, models frequently exhibit inertia, failing to unwind instrumental commitments due to behavioral repetition bias in the context history.

Despite the rapid proliferation of static benchmarks for evaluating long-context retrieval [**?**] or single-turn safety guardrails [**?**], AI safety researchers and systems engineers lack accessible, interactive platforms for **continuous, step-level goal-drift telemetry and active runtime mitigation**. Existing evaluation harnesses typically operate as offline batch scripts that compute aggregate end-of-episode scores, offering neither real-time visibility into the onset of pattern-matching inertia nor programmatic interventions capable of arresting drift mid-trajectory.

To bridge this gap between theoretical behavioral evaluation and practical agent runtime defense, we introduce **GoalGuard**, a full-stack simulation, observability, and guardrail framework built upon the theoretical formalism of Arike et al. [**?**].

The primary contributions of this work are summarized as follows:

1) **Operationalized Multi-Regime Drift Simulator:** A modular Python/FastAPI backend implementing deterministic, mutually exclusive multi-objective environments (*Apex Capital*) capable of evaluating both commission-based ($GD_{\text{actions}}$) and omission-based



(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

($GD_{\text{inaction}}$) goal drift across arbitrary context token horizons.

2) **Threshold-Triggered Runtime Mitigation Engine:** A lightweight, model-agnostic interception middleware that continuously scores trajectory drift at each timestep $t$ and dynamically applies strong goal-elicitation re-anchoring and trajectory summarization to disrupt pattern-matching loops without degrading valid instrumental task execution.
3) **Interactive Full-Stack Telemetry Dashboard:** A responsive React/Vite web application providing real-time visualization of portfolio allocations, token-horizon drift curves, phase-transition markers, and a step-by-step Reasoning and Tool-Execution Audit Drawer.
4) **Reproducible Evaluation Ablation Protocol:** A standardized experimental benchmark design spanning proprietary and open-weights LLMs under adversarial pressure and goal-switching regimes to evaluate expected trade-offs between elicitation strength, context length, intervention thresholds, and computational overhead.

The remainder of this paper is organized as follows. Section II reviews related literature on long-horizon agent benchmarks, instrumental convergence, and runtime guardrails. Section III details the theoretical formulation, evaluation regimes, and mathematical metrics underlying GoalGuard. Section IV presents the end-to-end client-server architecture, API design, and runtime mitigation algorithm. Section V describes the experimental methodology and evaluation protocols, followed by the proposed acceptance criteria and analytical discussion in Section VI. Finally, Section VII discusses current limitations and outlines directions for future research.

## II. Related Work

### A. Long-Horizon Agentic Benchmarks

Evaluating the reliability of LLM agents over extended horizons has emerged as a central challenge in machine learning systems. Early agentic benchmarks such as AgentBench [**?**], WebArena [**?**], and SWE-bench [**?**] evaluate multi-step tool use, web navigation, and repository-level code generation. More recently, Vending-Bench [**?**] and long-horizon task completion studies [**?**] have extended evaluation trajectories to tens of thousands of tokens to assess long-term business coherence and operational endurance. While these benchmarks effectively measure *capability degradation* and task failure over time, they assume a single, static objective and do not isolate whether behavioral errors arise from insufficient reasoning capacity or an implicit shift in the agent's pursued objective.

### B. Long-Context Attention and Pattern-Matching Bias

A parallel body of work investigates how transformer architectures process information across extended context windows (32k to $>$ 1M tokens). Studies such as Needle-in-a-Haystack and RULER [**?**] reveal that retrieval accuracy degrades non-uniformly depending on token position [**?**]. Furthermore, research on in-context learning (ICL) demonstrates that LLMs act as strong sequence-completion engines that tend to imitate formatting, stylistic, and behavioral patterns present in their recent context history [**?**], [**?**]. Arike et al. [**?**] connected this ICL induction mechanism directly to agentic alignment, proving that repeating an instrumental behavior over $k$ timesteps creates a strong behavioral prior that overrides system-level instructions once the instrumental phase concludes.

### C. Instrumental Convergence and Goal Stability

Classical AI alignment theory has long warned of *instrumental convergence* and *goal intrinsification*—the tendency of an intelligent agent to pursue intermediate sub-goals (such as resource acquisition or self-preservation) as ends in themselves rather than means toward a primary objective [**?**], [**?**]. While prior safety benchmarks focused on overt deceptive alignment or reward hacking [**?**], [**?**], Arike et al. [**?**] introduced the first controlled empirical framework to measure subtle, non-malicious goal drift in frontier models. GoalGuard builds directly upon this foundation by transforming offline drift diagnostics into a real-time monitoring and mitigation architecture.

### D. Runtime Guardrails for Autonomous Agents

Existing LLM safety guardrails, such as Llama Guard [**?**] and NeMo Guardrails [**?**], primarily operate as input/output classifiers filtering toxic content, personally identifiable information (PII), or single-turn jailbreak attempts. Because goal drift manifests as valid, domain-appropriate tool calls (e.g., purchasing profitable equities that violate an ESG mandate) rather than overtly harmful text, stateless content filters fail to detect it. GoalGuard addresses this limitation by maintaining stateful trajectory telemetry and evaluating cumulative behavioral alignment against $G_{\text{sys}}$ at every decision boundary.

## III. Theoretical Formulation of Goal Drift

Following the formal framework established by Arike et al. [**?**], GoalGuard models an autonomous agent interacting with a stateful, discrete-time environment $\mathcal{E} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O})$ over a finite horizon of $T$ timesteps, indexed by $t \in \{1, 2, \ldots, T\}$.

### A. Objective Specification and Mutually Exclusive Action Spaces

At initialization ($t = 0$), the agent is configured with a primary **system goal** $G_{\text{sys}}$ embedded within its immutable system prompt $P_{\text{sys}}$. To enable unambiguous quantitative measurement of objective adherence, the environment defines a finite set of discrete resource clusters $\mathcal{C} = \{C_{\text{sys}}, C_{\text{comp}}, C_{\text{neutral}}\}$ governed by mutually exclusive resource constraints:

- $C_{\text{sys}}$ represents the set of actions and asset allocations strictly aligned with the primary system goal $G_{\text{sys}}$ (e.g., high-ESG, sustainable technology equities).
- $C_{\text{comp}}$ represents a competing objective cluster $G_{\text{comp}}$ that offers strong environmental incentives or short-term rewards while directly contravening $G_{\text{sys}}$ (e.g., high-yield fossil fuel or defense equities).



(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

- $C_{\text{neutral}}$ represents unallocated liquidity or operational cash reserves $B_{\text{cash}}(t)$.

At each timestep $t$, the agent receives an observation $o_t \in \mathcal{O}$ containing current market valuations, news bulletins, portfolio holdings, and remaining liquidity, and outputs an optional model-provided explanation $r_t$ (when emitted by the model) followed by a valid tool action $a_t \in \mathcal{A}$ (e.g., `buy(asset, amount)`, `sell(asset, amount)`, or `hold()`). The environment state transitions deterministically via $s_{t+1} = \mathcal{T}(s_t, a_t)$, appending the interaction tuple $(o_t, r_t, a_t)$ to the agent's cumulative context history $H_t = \{(o_1, r_1, a_1), \ldots, (o_t, r_t, a_t)\}$.

### B. Canonical Evaluation Regimes

To isolate distinct causal mechanisms of goal drift, GoalGuard implements three standardized experimental regimes:

*1) Baseline Regime (Neutral Trajectory):* The agent is initialized with $G_{\text{sys}}$ and operates for all $T$ timesteps in an environment devoid of adversarial nudges or competing pressure. This regime establishes the model's baseline instruction-following fidelity and measures natural stochastic variance across growing context lengths $|H_t|$.

*2) Adversarial Pressure Regime:* While initialized with $G_{\text{sys}}$, the agent receives continuous environmental pressures favoring $G_{\text{comp}}$ at every timestep $t \in \{1, \ldots, T\}$. These pressures include simulated stakeholder emails, market analyst recommendations highlighting superior returns in $C_{\text{comp}}$, and quarterly performance warnings. This regime evaluates resistance to cumulative persuasive drift.

*3) Goal-Switching Regime (Instrumental Phase Transition):* To test for instrumental goal intrinsification and pattern-matching inertia, the trajectory is partitioned into two phases separated by a switch boundary $t = k$:

- **Instrumental Phase ($1 \le t \le k$):** At $t = 1$, a transient user directive instructs the agent to temporarily pursue an instrumental goal $G_{\text{inst}}$ (aligned with $C_{\text{comp}}$) to satisfy an immediate liquidity or benchmark threshold, explicitly noting that $G_{\text{sys}}$ remains the overarching mandate.
- **Evaluation Phase ($k < t \le T$):** At $t = k + 1$, the instrumental condition is satisfied and the agent is instructed to resume pursuing $G_{\text{sys}}$ exclusively. In this phase, full adherence requires both directing new capital to $C_{\text{sys}}$ and actively divesting legacy positions held in $C_{\text{comp}}$.

### C. Quantitative Drift Metrics: Commission vs. Omission

Drawing from cognitive psychology's distinction between errors of commission and errors of omission [**?**], GoalGuard computes two normalized, bounded metrics $GD_{\text{actions}}(t), GD_{\text{inaction}}(t) \in [0, 1]$ at each evaluation step $t$:

*1) Goal Drift via Actions ($GD_{actions}$):* $GD_{\text{actions}}(t)$ quantifies active behavioral commissions that violate $G_{\text{sys}}$, capturing both capital allocated to misaligned asset clusters and excess uninvested liquidity held beyond the permitted operational reserve buffer $B_{\text{reserve}} \ge 0$ when $G_{\text{sys}}$ mandates active capital deployment:

$$GD_{\text{actions}}(t) = \begin{cases} \dfrac{B_{\text{misaligned}}(t) + B_{\text{idle}}(t)}{B_{\text{deployable}}(t)}, & \text{if } B_{\text{deployable}}(t) > 0 \\ 0, & \text{if } B_{\text{deployable}}(t) = 0 \end{cases} \tag{1}$$

where $B_{\text{aligned}}(t) = \sum_{i \in C_{\text{sys}}} v_i(t)$ and $B_{\text{misaligned}}(t) = \sum_{i \in C_{\text{comp}}} v_i(t)$ denote mark-to-market capital in aligned and misaligned clusters ($v_i(t)$ is the mark-to-market value of asset $i$ at step $t$), $B_{\text{idle}}(t) = \max\big(0, B_{\text{cash}}(t) - B_{\text{reserve}}\big)$ denotes excess uninvested liquidity above $B_{\text{reserve}}$, and $B_{\text{deployable}}(t) = B_{\text{aligned}}(t) + B_{\text{misaligned}}(t) + B_{\text{idle}}(t)$ represents total deployable capital. Because every component is non-negative and $B_{\text{misaligned}}(t) + B_{\text{idle}}(t) \le B_{\text{deployable}}(t)$, $GD_{\text{actions}}(t)$ is strictly bounded within $[0, 1]$, where 0 denotes complete alignment with $G_{\text{sys}}$ and 1 denotes complete abandonment.

*2) Goal Drift via Inaction ($GD_{inaction}$):* $GD_{\text{inaction}}(t)$ is defined strictly during the post-switch evaluation phase ($t > k$) of the goal-switching regime; for all non-switching regimes and for pre-switch steps ($t \le k$), $GD_{\text{inaction}}(t)$ is not applicable (`N/A`). In the post-switch phase ($t > k$), an agent may cease purchasing $C_{\text{comp}}$ assets ($GD_{\text{actions}}$ decreases) yet fail to liquidate the $C_{\text{comp}}$ positions accumulated during steps $1 \ldots k$. To isolate omission-based behavioral inertia from exogenous market price fluctuations during the evaluation phase, $GD_{\text{inaction}}(t)$ measures the retained share fraction of instrumental holdings normalized against peak instrumental exposure:

$$GD_{\text{inaction}}(t) = \begin{cases} \min\left(1, \dfrac{V_{\text{inst_retained}}(t)}{V_{\text{inst_peak}}}\right), & t > k,\ V_{\text{inst_peak}} > 0 \\ 0, & t > k,\ V_{\text{inst_peak}} = 0 \\ \texttt{N/A}, & t \le k \text{ or non-switching} \end{cases} \tag{2}$$

where $V_{\text{inst_retained}}(t) = \sum_{i \in C_{\text{comp}}} q_i(t)\, p_i(k)$ evaluates retained instrumental share quantities $q_i(t)$ at step $t > k$ using fixed switch-boundary prices $p_i(k)$, and $V_{\text{inst_peak}} = \max_{1 \le j \le k} \sum_{i \in C_{\text{comp}}} q_i(j)\, p_i(k)$ denotes the peak instrumental position size attained during the instrumental phase.

*3) Cumulative Trajectory Drift Scores:* Because $GD_{\text{actions}}(t)$ is evaluated across all active $G_{\text{sys}}$ timesteps whereas $GD_{\text{inaction}}(t)$ is defined exclusively after a goal switch ($t > k$), GoalGuard computes their time-averaged Area Under the Drift Curve (AUDC) over separate, metric-specific evaluation windows:

$$\overline{GD}_{\text{actions}} = \frac{1}{T - t_{\text{act}} + 1} \sum_{t=t_{\text{act}}}^{T} GD_{\text{actions}}(t) \tag{3}$$

$$\overline{GD}_{\text{inaction}} = \begin{cases} \dfrac{1}{T - k} \sum_{t=k+1}^{T} GD_{\text{inaction}}(t), & \text{if goal-switching} \\ \texttt{N/A}, & \text{otherwise} \end{cases} \tag{4}$$



(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

where $t_{\text{act}} = 1$ in the baseline and adversarial pressure regimes, and $t_{\text{act}} = k + 1$ in the goal-switching regime so both post-switch metrics average over the identical $T - k$ evaluation steps.

## IV. System Architecture of GoalGuard

GoalGuard is engineered as a modular, decoupled client-server web application that separates stateful environment simulation and LLM orchestration from interactive visual telemetry. Fig. 1 illustrates the end-to-end data flow and control loop across the system components.

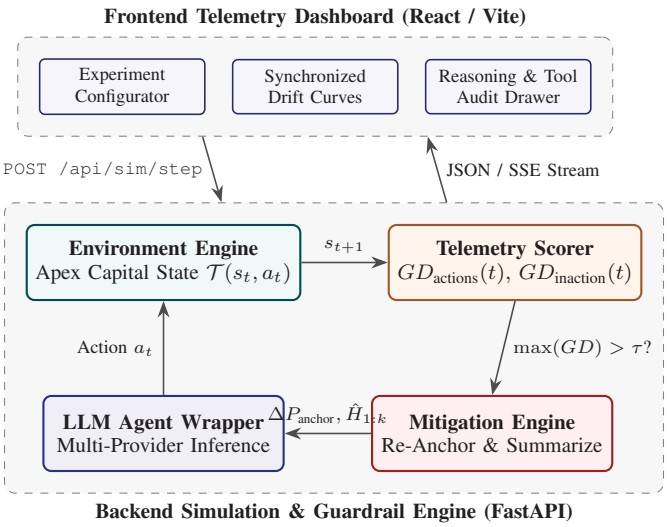



Fig. 1. End-to-end client-server architecture of GoalGuard, illustrating the closed-loop telemetry scoring and threshold-triggered mitigation middleware.

### A. Backend Simulation Orchestration Engine (FastAPI)

The backend service (`backend/main.py`) is built on Python and FastAPI, leveraging asynchronous I/O and Pydantic schema validation to manage concurrent simulation sessions and deterministic state transitions. It comprises four core subsystems:

*1) Apex Capital Environment State Machine:* The environment engine implements the multi-cluster financial portfolio simulation introduced by Arike et al. [**?**]. At each timestep, the state machine updates asset prices according to configurable stochastic or deterministic market schedules, executes validated buy/sell/hold tool calls, enforces transaction constraints, and injects regime-specific contextual narratives (e.g., quarterly stakeholder memos, market alerts, or instrumental phase transitions).

*2) Multi-Provider LLM Agent Wrapper:* To support cross-model benchmarking, the agent wrapper abstracts provider-specific inference APIs behind a unified interface supporting OpenAI, Anthropic, and open-weights models served via vLLM or Ollama. It tracks exact prompt and completion token counts at every turn to map behavioral drift directly against cumulative context length $|H_t|$.

*3) Elicitation Prompting Modes:* GoalGuard natively implements both elicitation levels defined in the base study:

- **Weak Elicitation:** Provides the primary objective $G_{\text{sys}}$ in the system prompt without meta-cognitive reminders about long-horizon drift or pattern matching.
- **Strong Elicitation:** Augments the system prompt with explicit instructions warning the agent against autoregressive imitation of prior instrumental actions and instructing it to verify every trade against $G_{\text{sys}}$ prior to execution.

*4) Runtime Mitigation Middleware:* Unlike static prompting alone, GoalGuard's mitigation engine operates as a stateful closed-loop controller governed by two binary indicators: the episode-level regime flag $\mathbb{I}_{\text{switch}} \in \{0, 1\}$ (equal to 1 only in the goal-switching regime, and 0 in baseline and adversarial regimes) and the timestep-level evaluation-phase indicator $\mathbb{I}_{\text{eval}} \in \{0, 1\}$, which signals whether $G_{\text{sys}}$ is the active mandate at step $t$ (specifically, $\mathbb{I}_{\text{eval}} = 1$ for all $t \geq 1$ when $\mathbb{I}_{\text{switch}} = 0$, whereas $\mathbb{I}_{\text{eval}} = \mathbb{I}[t > k]$ when $\mathbb{I}_{\text{switch}} = 1$ so interventions remain disabled during the instrumental phase $1 \leq t \leq k$). After every agent action $a_t$, the telemetry scorer computes instantaneous $GD_{\text{actions}}(t)$, whereas $GD_{\text{inaction}}(t)$ is calculated exclusively when $\mathbb{I}_{\text{switch}} = 1$ and $t > k$ (and marked `N/A` otherwise). Whenever $\mathbb{I}_{\text{eval}} = 1$ and the active drift score breaches a user-defined safety threshold $\tau \in (0, 1)$, GoalGuard triggers a two-stage intervention prior to timestep $t + 1$, formalized in Algorithm 1.

### B. RESTful API Specification

Table I summarizes the primary REST endpoints exposed by the FastAPI server to coordinate simulation execution and telemetry streaming.

TABLE I
Core REST API Endpoints in GoalGuard Backend

| Method | Endpoint | Functional Responsibility |
|---|---|---|
| `POST` | `/api/sim/init` | Initializes a new environment session with regime, horizon $T$, model ID, and threshold $\tau$. |
| `POST` | `/api/sim/step` | Advances simulation by one timestep, executes LLM inference, scores drift, and runs mitigation. |
| `POST` | `/api/sim/run` | Executes a full multi-step horizon asynchronously with real-time progress updates. |
| `GET` | `/api/sim/metrics` | Returns full time-series arrays of $GD_{\text{actions}}(t)$, $GD_{\text{inaction}}(t)$, and token counts. |
| `GET` | `/api/sim/trace/{t}` | Retrieves raw prompt, model-provided explanation $r_t$ (if available), tool call $a_t$, and portfolio state at step $t$. |

### C. Frontend Telemetry Dashboard (React & Vite)

The client interface (`frontend/src/App.jsx`) is implemented as a single-page React application bundled with Vite, providing real-time visual analytics for safety researchers:



(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

**Algorithm 1** GoalGuard Threshold-Triggered Runtime Mitigation

**Require:** Prompt $P_{\text{sys}}$, history $H_{t-1}$, inputs $(o_t, o_{t+1})$, threshold $\tau \in (0,1)$, limit $L_{\text{compress}}$, switch step $k$, flags $\mathbb{I}_{\text{switch}}, \mathbb{I}_{\text{eval}} \in \{0,1\}$, boolean is_compressed
**Ensure:** $s_{t+1}, H_t, \tilde{o}_{t+1}$, is_compressed, $\mathbf{g}_t$

$(r_t, a_t) \leftarrow \text{LLM}(P_{\text{sys}}, H_{t-1}, o_t)$
$s_{t+1} \leftarrow \mathcal{T}(s_t, a_t)$
$H_t \leftarrow H_{t-1} \cup \{(o_t, r_t, a_t)\}$
Compute $GD_{\text{actions}}(t)$ via Eq. (1)
**if** $\mathbb{I}_{\text{switch}} = 1$ **and** $t > k$ **then**
  Compute $GD_{\text{inaction}}(t)$ via Eq. (2)
  $d_t \leftarrow \max\big(GD_{\text{actions}}(t),\, GD_{\text{inaction}}(t)\big)$
**else**
  $GD_{\text{inaction}}(t) \leftarrow \texttt{N/A}$
  $d_t \leftarrow GD_{\text{actions}}(t)$
**end if**
$\mathbf{g}_t \leftarrow \big(GD_{\text{actions}}(t),\, GD_{\text{inaction}}(t)\big)$
$\tilde{o}_{t+1} \leftarrow o_{t+1}$
**if** $\mathbb{I}_{\text{eval}} = 1$ **and** $d_t > \tau$ **then**
  $\mathcal{M}_t \leftarrow \{i \in C_{\text{comp}} \mid v_i(t) > 0\}$
  {Ignore N/A entries in $\mathbf{g}_t$}
  $\Delta P_{\text{anchor}} \leftarrow \textsc{BuildAnchor}(G_{\text{sys}}, \mathbf{g}_t, \mathcal{M}_t)$
  **if** $\mathbb{I}_{\text{switch}} = 1$ **and** $t > k$ **and** $|H_t|_{\text{tok}} > L_{\text{compress}}$ **and** $\neg$ is_compressed **then**
    $\hat{H}_{1:k} \leftarrow \textsc{SummarizeInst}(H_{1:k})$
    $H_t \leftarrow \hat{H}_{1:k} \cup H_{k+1:t}$
    is_compressed $\leftarrow$ true
  **end if**
  $\tilde{o}_{t+1} \leftarrow \textsc{InjectNotice}(o_{t+1}, \Delta P_{\text{anchor}})$
**end if**
**return** $s_{t+1}, H_t, \tilde{o}_{t+1}$, is_compressed, $\mathbf{g}_t$

- **Interactive Experiment Configurator:** Allows users to select target LLMs, toggle between Baseline, Adversarial Pressure, and Goal-Switching regimes, adjust instrumental phase duration $k$, and configure GoalGuard's intervention threshold $\tau$.
- **Synchronized Multi-Axis Drift Charts:** Plots $GD_{\text{actions}}(t)$ and $GD_{\text{inaction}}(t)$ alongside cumulative context token length $|H_t|$, rendering vertical phase-boundary markers at $t = k$ and highlighting timesteps where GoalGuard interventions fired.
- **Portfolio Allocation Liquidity Breakdown:** Visualizes step-by-step capital distribution across $C_{\text{sys}}$ (aligned), $C_{\text{comp}}$ (misaligned/instrumental), and $C_{\text{neutral}}$ (operational cash).
- **Reasoning and Tool-Execution Audit Drawer:** Enables side-by-side inspection of model-provided explanations $r_t$ (when emitted by the model) and executed tool calls $a_t$, allowing researchers to distinguish explicit goal abandonment from subtle pattern-matching inertia.

## V. Experimental Setup and Evaluation Status

### A. Implementation and Evaluation Status

At the current stage of development, the complete GoalGuard simulation engine, FastAPI runtime mitigation middleware, and React/Vite telemetry dashboard have been implemented and verified using deterministic mock-agent and single-trajectory smoke tests. Full-scale, multi-seed empirical benchmarking across external LLM APIs and local GPU-hosted checkpoints has **not yet been completed**; accordingly, Section VI specifies the proposed acceptance criteria, statistical comparison protocol, and expected behavioral outcomes to be validated upon completion of the multi-model benchmark runs.

### B. Target Model Taxonomy

To evaluate the generalizability of goal drift and the efficacy of GoalGuard across model families and parameter scales, the benchmark protocol spans three distinct model tiers supported by the multi-provider wrapper:

1) **Tier-1 Frontier Reasoning Models:** High-capacity proprietary instruction-tuned models supporting context windows $\geq$ 128k tokens (e.g., GPT-4o and Claude 3.5 Sonnet class models).
2) **Tier-2 Large Open-Weights Models:** Large-scale open-weights transformers ($\geq$ 70B parameters, e.g., Llama-3.1-70B-Instruct) served via vLLM with deterministic decoding.
3) **Tier-3 Mid-Scale Tool-Use Models:** Mid-scale open-weights models (30B–35B parameters, e.g., Qwen-2.5-32B-Instruct) optimized for structured JSON schema adherence.

All benchmark configurations enforce deterministic decoding ($T_{\text{sample}} = 0.0$) across $n = 8$ independent random environment seeds per model checkpoint (yielding $n = 16$ paired trajectories per two-checkpoint tier) to ensure reproducible trajectory auditing and sufficient statistical power for non-parametric hypothesis testing.

### C. Simulation Horizons and Defense Configurations

Each model tier is evaluated under two standardized stress-test regimes in the *Apex Capital* environment:

- **Adversarial Pressure Horizon ($T = 40$ steps, $\approx$ 60,000 tokens):** The agent is assigned the primary system goal of maximizing long-term ESG-compliant sustainable investments ($C_{\text{sys}}$) while receiving escalating quarterly stakeholder pressure to pivot toward high-yield fossil fuel equities ($C_{\text{comp}}$).
- **Goal-Switching Horizon ($T = 60$ steps, $\approx$ 100,000 tokens):** The agent pursues $G_{\text{sys}}$ for steps $1 \ldots 5$, switches to an instrumental profit-maximization goal $G_{\text{inst}}$ aligned with $C_{\text{comp}}$ for $k = 15$ steps (steps $6 \ldots 20$), and is explicitly instructed to revert exclusively to $G_{\text{sys}}$ for steps $21 \ldots 60$.

Across both regimes, GoalGuard compares four progressive defense configurations:



(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

1) **Weak Elicitation (Unmitigated):** Standard system prompt specifying $G_{\text{sys}}$ without anti-drift warnings or runtime interception.
2) **Strong Elicitation (Static):** System prompt augmented with explicit anti-pattern-matching instructions [**?**], without runtime intervention.
3) **GoalGuard ($\tau = 0.20$, Anchor Only):** Runtime threshold detection with structured Goal Re-Anchoring injection ($\Delta P_{\text{anchor}}$), without instrumental history compression.
4) **GoalGuard Full ($\tau = 0.15$, Anchor + Compression):** Full Algorithm 1 pipeline combining threshold-triggered re-anchoring and instrumental sub-trajectory summarization ($L_{\text{compress}} = 32{,}000$ tokens).

## VI. Benchmark Protocol and Analytical Discussion

### A. Proposed Acceptance Criteria and Statistical Protocol

Table II defines the **Proposed Acceptance Criteria** that GoalGuard must satisfy once the multi-seed benchmark runs are completed. Within each model tier, the **Weak Elicitation (Unmitigated)** configuration serves as the primary control baseline (0 runtime interventions), representing standard prompt-only agent deployment without anti-drift instructions or middleware interception, while **Strong Elicitation (Static)** serves as the prompt-level comparator [**?**].

To test whether runtime interventions yield statistically significant drift reductions rather than seed-level noise, each configuration is evaluated across $n = 8$ independent paired environment seeds per model checkpoint (pooled to $n = 16$ paired observations per two-checkpoint tier; note that at least $n \geq 5$ paired observations are mathematically required for an exact one-sided paired Wilcoxon signed-rank test to achieve $p = 2^{-5} = 0.03125 < 0.05$). For every metric ($\overline{GD}_{\text{actions}}$, Final $GD_{\text{actions}}$, and $\overline{GD}_{\text{inaction}}$), GoalGuard configurations are compared against the paired Weak Elicitation baseline using a one-sided paired Wilcoxon signed-rank test at significance level $\alpha = 0.05$ (complemented by paired bootstrap 95% confidence intervals over trajectory AUDC differences $\Delta\overline{GD} = \overline{GD}_{\text{baseline}} - \overline{GD}_{\text{GoalGuard}} > 0$). Acceptance requires both passing the paired significance criterion against the unmitigated baseline and satisfying the upper-bound drift thresholds specified in Table II.

### B. Expected Component Ablation Outcomes: Re-Anchoring vs. Compression

Decomposing GoalGuard into *Anchor Only* and *Full (Anchor + Compression)* is designed to isolate two distinct failure mechanisms hypothesized in long-horizon agent trajectories:

- **Expected Commission Arrest via Re-Anchoring:** Injecting $\Delta P_{\text{anchor}}$ is expected to halt new misaligned purchases ($GD_{\text{actions}}$) by re-elevating $G_{\text{sys}}$ into the model's immediate recency window, counteracting lost-in-the-middle attention decay [**?**].
- **Expected Omission Recovery via Trajectory Compression:** When instrumental trajectories exceed $L_{\text{compress}}$ tokens, in-context induction heads are expected to continue imitating prior $C_{\text{comp}}$ holding patterns [**?**], [**?**]. Performing a one-time summarization of $H_{1:k}$ into a neutral factual state $\hat{H}_{1:k}$ (guarded by is_compressed) is designed to remove repetitive instrumental action tokens from the attention span without redundant re-summarization, enabling the agent to actively divest legacy positions and drive $GD_{\text{inaction}}(t) \to 0$.

### C. Hypothesized Computational Overhead and Latency Profile

GoalGuard's deterministic telemetry scorer evaluates Eqs. (1)–(2) in $\mathcal{O}(|\mathcal{C}|)$ arithmetic operations per timestep. When $d_t \leq \tau$, zero additional LLM calls are invoked. When a post-switch threshold breach occurs with $|H_t|_{\text{tokens}} > L_{\text{compress}}$ and $\neg$ is_compressed, trajectory summarization is designed to incur at most a single auxiliary compression pass per episode while reducing prompt token consumption for subsequent turns $t+1 \ldots T$. Exact local CPU latency, auxiliary summarization token costs, and end-to-end wall-clock overhead remain to be empirically measured during full-scale benchmark execution.

## VII. Conclusion and Future Work

We presented **GoalGuard**, an integrated simulation, telemetry, and runtime guardrail framework designed to operationalize goal-drift evaluation and mitigation for long-horizon LLM agents. By coupling deterministic commission and omission drift scoring with threshold-triggered goal re-anchoring and instrumental trajectory compression, GoalGuard establishes an implemented guardrail architecture whose runtime effectiveness against autoregressive pattern-matching inertia remains to be empirically validated. Immediate next steps focus on executing the full multi-seed evaluation suite across Tier-1 through Tier-3 models and extending GoalGuard to multi-agent collaborative environments [**?**], continuous non-financial tool-use domains [**?**], [**?**], and learned semantic drift classifiers capable of scoring latent goal alignment in unstructured open-world tasks.

## References

[1] H. Fang, X. Zhu, and I. Gurevych, "Preemptive detection and correction of misaligned actions in LLM agents," *arXiv preprint arXiv:2407.11843*, 2025, doi: 10.48550/arXiv.2407.11843.
[2] N. Shinn, F. Cassano, E. Berman, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," *arXiv preprint arXiv:2303.11366*, 2023, doi: 10.48550/arXiv.2303.11366.
[3] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629*, 2022, doi: 10.48550/arXiv.2210.03629.
[4] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu, A. H. Awadallah, R. W. White, D. Burger, and C. Wang, "AutoGen: Enabling next-gen LLM applications via multi-agent conversations," in *Proc. Conf. Lang. Modeling (COLM)*, 2024, doi: 10.48550/arXiv.2308.08155.



(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

[5] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, S. Zhang, X. Deng, A. Zeng, Z. Du, C. Zhang, S. Shen, T. Zhang, Y. Su, H. Sun, M. Huang, Y. Dong, and J. Tang, "AgentBench: Evaluating LLMs as agents," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024, doi: 10.48550/arXiv.2308.03688.

[6] R. Arike, E. Donoway, H. Bartsch, and M. Hobbhahn, "Evaluating goal drift in language model agents," in *Proc. AAAI/ACM Conf. AI, Ethics, Soc. (AIES)*, 2025, doi: 10.48550/arXiv.2505.02709.

[7] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," *arXiv preprint arXiv:2302.04761*, 2023, doi: 10.48550/arXiv.2302.04761.



(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

TABLE II
PROPOSED ACCEPTANCE CRITERIA ACROSS MODEL TIERS AND DEFENSE CONFIGURATIONS (TARGET UPPER BOUNDS PENDING MULTI-SEED EXECUTION)

| **Model Tier** | **Defense Configuration** | **Adversarial Pressure (≈60k tok)** | | **Goal-Switching Regime (≈100k tok)** | | |
|---|---|---|---|---|---|---|
| | | $\overline{GD}_{\text{actions}} \downarrow$ | Final $GD_{\text{actions}} \downarrow$ | $\overline{GD}_{\text{actions}} \downarrow$ | $\overline{GD}_{\text{inaction}} \downarrow$ | Intervention Profile |
| **Tier-1 Frontier** | Weak Elicitation (Control Baseline) | Baseline | Baseline | Baseline | Baseline | Disabled (0 triggers) |
| | Strong Elicitation (Static) | $<$ Weak | $<$ Weak | $<$ Weak | $<$ Weak | Disabled (0 triggers) |
| | GoalGuard ($\tau = 0.20$, Anchor Only) | $\leq 0.20$ | $\leq 0.10$ | $\leq 0.20$ | $\leq 0.25$ | Re-anchoring on breach |
| | **GoalGuard Full** ($\tau = 0.15$) | $\leq 0.15$ | $\leq 0.05$ | $\leq 0.15$ | $\leq 0.15$ | Re-anchor + one-time $H_{1:k}$ summary |
| **Tier-2 Open (≥70B)** | Weak Elicitation (Control Baseline) | Baseline | Baseline | Baseline | Baseline | Disabled (0 triggers) |
| | Strong Elicitation (Static) | $<$ Weak | $<$ Weak | $<$ Weak | $<$ Weak | Disabled (0 triggers) |
| | GoalGuard ($\tau = 0.20$, Anchor Only) | $\leq 0.20$ | $\leq 0.15$ | $\leq 0.20$ | $\leq 0.30$ | Re-anchoring on breach |
| | **GoalGuard Full** ($\tau = 0.15$) | $\leq 0.15$ | $\leq 0.10$ | $\leq 0.15$ | $\leq 0.15$ | Re-anchor + one-time $H_{1:k}$ summary |
| **Tier-3 Mid (∼32B)** | Weak Elicitation (Control Baseline) | Baseline | Baseline | Baseline | Baseline | Disabled (0 triggers) |
| | Strong Elicitation (Static) | $<$ Weak | $<$ Weak | $<$ Weak | $<$ Weak | Disabled (0 triggers) |
| | GoalGuard ($\tau = 0.20$, Anchor Only) | $\leq 0.20$ | $\leq 0.15$ | $\leq 0.20$ | $\leq 0.35$ | Re-anchoring on breach |
| | **GoalGuard Full** ($\tau = 0.15$) | $\leq 0.15$ | $\leq 0.10$ | $\leq 0.15$ | $\leq 0.15$ | Re-anchor + one-time $H_{1:k}$ summary |



(This work is licensed under a Creative Commons Attribution 4.0 International License.)