

SKYNET: Workflow-Level Anomaly Detection for Agentic AI via Semantic and Structural Modeling

Chaoyu Zhang
Virginia Tech
Arlington, VA, USA
chaoyu@vt.edu

Hexuan Yu
Virginia Tech
Arlington, VA, USA
hexuanyu@vt.edu

Heng Jin
Virginia Tech
Arlington, VA, USA
hengj@vt.edu

Shanghao Shi
WashU
St. Louis, MO, USA
shanghao@wustl.edu

Ning Zhang
WashU
St. Louis, MO, USA
zhang.ning@wustl.edu

Yi Shi
Virginia Tech
Arlington, VA, USA
yshi@vt.edu

Yulia R. Gel
Virginia Tech
Blacksburg, VA, USA
ygl@vt.edu

Y. Thomas Hou
Virginia Tech
Blacksburg, VA, USA
hou@vt.edu

Wenjing Lou
Virginia Tech
Arlington, VA, USA
wjlu@vt.edu

Abstract

Agentic AI systems execute complex tasks through long-horizon workflows of planning, tool use, and multi-agent coordination. Task failures in these systems often originate from a single step, such as an injected prompt or a flawed plan, and are then amplified through downstream dependencies as the corrupted step propagates across many subsequent agents and tool calls. Existing defenses either target a specific class of attacks or failures, or inspect individual prompts and steps in isolation. Both leave the global dependency structure of a workflow unexamined, and miss the inconsistencies that only emerge when the execution is viewed as a whole. We argue that anomaly detection for agentic AI must reason at the workflow level, where global execution structure exposes signals that local checks cannot see. We present SKYNET, a principled workflow-level anomaly detection framework that turns observed multi-agent execution into directed workflow graphs and scores them against learned benign behavior. SKYNET jointly models the semantic execution context and the structural organization of inter-agent delegation, tool invocation, and data-flow dependencies, and trains only on benign workflows. Because training never sees attacks or failures, this design naturally extends to zero-day detection: any execution that violates benign workflow regularities surfaces as off-manifold geometry under a single decision rule. We evaluate SKYNET on three public agentic safety and failure benchmarks. It sustains high recall together with a sub-1% false positive rate, with per-workflow and per-step latencies low enough for online monitoring of agentic AI runtimes.

CCS Concepts

• **Security and privacy** → **Intrusion/anomaly detection and malware mitigation**; *Distributed systems security*; • **Computing methodologies** → *Neural networks*; *Multi-agent systems*.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MobiHoc '26, Tokyo, Japan

© 2026 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
<https://doi.org/10.1145/nnnnnnn.nnnnnnn>

Keywords

Agent Security, Agent Guardrail, Workflow-Level Anomaly Detection, Graph-Based Representation Learning

ACM Reference Format:

Chaoyu Zhang, Hexuan Yu, Heng Jin, Shanghao Shi, Ning Zhang, Yi Shi, Yulia R. Gel, Y. Thomas Hou, and Wenjing Lou. 2026. SKYNET: Workflow-Level Anomaly Detection for Agentic AI via Semantic and Structural Modeling. In *Proceedings of The 27th International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc '26)*. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

1 Introduction

Modern agentic AI stacks such as OpenAI’s ChatGPT Agent [24], Anthropic’s orchestrator-worker patterns [4], Microsoft’s AutoGen [38], and Google’s Agent-to-Agent (A2A) protocol [12] compose specialized large language model (LLM) agents that plan, invoke external tools through the Model Context Protocol (MCP) [5, 25], delegate sub-tasks, and exchange artifacts over long horizons [27, 32, 34, 38, 41, 43]. The system-level behavior of these stacks is best characterized by the *workflows* they realize, which are ordered, dependency-rich execution structures over planning steps, tool calls, inter-agent messages, and system events, together with the temporal and causal links that determine how each step shapes the rest of the execution.

This shift from single-call inference to long-horizon agentic execution introduces a qualitatively new *risk surface* that subsumes both unintentional failures and adversarial threats under a single operational concern. The threats we observe on this surface come from two complementary sources. *Intrinsic workflow failures* arise from the system itself, including flawed planning, ambiguous user instructions, hallucinated intermediates, and breakdowns in tool use or coordination [8, 19, 44, 49]. *Adversarial workflow manipulations* arise from intentional interference such as prompt injection, tool-description injection, privilege escalation, or laundering of malicious intent across multiple agents [9, 14, 23, 28, 45]. Despite these very different origins, both classes typically begin from a single perturbed step, such as a benign-yet-erroneous user request, a flawed plan, or a misjudged observation on one side, or a malicious instruction or tampered tool description on the other. Once such a step is produced, the workflow keeps consuming and propagating it: downstream

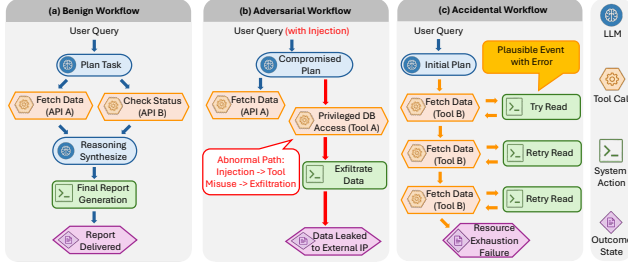


Figure 1: Agentic AI execution workflows under normal and abnormal conditions. (a) Benign workflow with coherent planning, tool use, and execution. (b) Adversarial manipulation, where injected steps redirect execution. (c) Intrinsic failure, where non-adversarial errors (e.g., misjudged progress, retries) accumulate.

agents condition on it, tools receive it, and later decisions are made on top of it, so the original perturbation is amplified across the rest of the execution. As a result, both surface as abnormal execution trajectories that diverge from intended task behavior (Figure 1).

Existing approaches mitigate this risk by attaching defenses to specific layers of the agentic stack. One line restricts how agents handle untrusted inputs [8], enforces execution-time invariants [35], or applies transactional protocols where formal specifications are available [19]. Another line contains compromised components through sandboxing [39], mediated tool access [17], and rule-based action verification [7]. A third line matches execution traces against patterns associated with known attack and failure classes [18, 21, 49]. Each of these is anchored to a specific class of problems or operates at the granularity of an individual prompt, action, or trace fragment, leaving the global dependency structure of a workflow only indirectly examined. As production agentic systems continuously add new tools, agents, and task domains, this leaves a long tail of cross-step anomalies that no single specification or pattern catches.

An important research question raised by this gap is therefore: *at what granularity should anomalies in agentic AI be detected?* Network intrusion detection went through the same transition. Early packet-based detectors flagged each packet in isolation and missed attacks that only became visible across a long session, which motivated record-based intrusion detection that aggregates packets into flows and reasons about them as a whole. Agentic AI is in the same position. Inspecting a single prompt, action, or trace fragment throws away the dependency structure that an injected or flawed step uses to spread, and the most informative signals are non-local, including role-inconsistent delegations, atypical tool-call pathways, and abnormal artifact-to-agent influence. A monitor for agentic AI must therefore reason over whole workflows, model both their semantics and their dependency structure, and remain agnostic to which particular attack or failure mode is in play.

We realize this view in SKYNET, a principled workflow-level anomaly detection framework for agentic AI. SKYNET plugs into the observation layer of modern agentic runtimes by consuming the structured execution records they already expose, and consolidates them into a directed workflow graph that captures planning, tool invocations, inter-agent delegation, data flow, and system events with

their temporal and causal dependencies. On top of this representation, SKYNET learns from benign workflows alone, jointly modeling *what* happens (semantic execution context) and *how* effects propagate (dependency structure), with training views built around the structural regularities of benign agentic execution rather than from random graph perturbation. The detector is therefore aligned with the benign workflow manifold, and exposes both threat classes as off-manifold geometry under a single decision rule. Because training never sees attacks or failures, this design naturally extends to *zero-day* detection: previously unseen attacks and novel failure modes that violate benign workflow regularities surface through the same off-manifold geometry, with no need to anticipate them.

We evaluate SKYNET on three public agentic benchmarks targeting complementary failure regimes: Agent-SafetyBench [48] (adversarial manipulations), AgentErrorBench [50] (intrinsic execution failures), and ATBench [20] (broad safety-in-the-wild). Training uses only benign trajectories from ETO [30] and never observes adversarial or failing workflows.

2 System Model, Threat Model, and Problem

2.1 System Model

Target agentic AI systems. We target modern multi-agent AI systems that execute complex user intents through structured, long-horizon task execution: orchestrator-worker architectures [4], conversational multi-agent frameworks such as Microsoft AutoGen [38] and OpenAI’s ChatGPT Agent [24], agent-to-agent interoperable ecosystems built on Google’s A2A protocol [12], and ReAct-style planners [32, 43] that invoke tools through the Model Context Protocol (MCP) [5, 25]. A common pattern recurs across these stacks: a planner (or orchestrator) agent decomposes a user task into subtasks, delegates them to specialized worker agents, and composes intermediate results through tool calls and inter-agent messages. The planner naturally acts as an *observer* of the execution it initiates, since the runtime already exposes structured traces of what each agent did, in what order, with which tools, and with what intermediate results. SKYNET plugs directly into this observation layer as a passive consumer of execution records that already exist for telemetry, debugging, and compliance; it does not modify agent logic, tool backends, inter-agent protocols, or LLM weights.

Observed execution records. Our system model focuses on *observed execution workflows* rather than the internal planning states of individual agents. We assume access to four classes of records standard across modern agentic platforms: (i) *planning and reasoning traces* (ReAct-style thought/action/observation tuples, orchestrator plans, intermediate scratchpads); (ii) *tool invocations* (MCP-style calls with typed inputs/outputs, calling agent identity, status); (iii) *inter-agent messages* through protocols such as A2A or AutoGen’s conversation primitives, including delegation requests, replies, and shared artifacts; and (iv) *system-level events* (retries, errors, sandbox decisions, timeouts). When a planner delegates a task, it monitors the resulting execution and collects these records within its visibility scope; nested delegations record evidence locally and return it on completion, so workflow evidence is propagated bottom-up and consolidated at the root planner.

Workflow construction. Given the consolidated record of an execution, SKYNET builds a directed workflow graph (Sec. 3) in a

runtime-agnostic way. Nodes correspond to observed execution steps annotated with structured attributes that summarize *what the step was*, including actor role, action type, tool/endpoint identifier, and status. Typed edges capture *how steps relate*: temporal precedence, delegation between agents, data-flow dependencies, and cross-agent communication. This construction is intentionally protocol-agnostic and treats platform-specific record formats as instantiations of the same abstraction; for example, an AutoGen conversation, an A2A-mediated interaction, a ChatGPT Agent trace, or a general MCP-tool workflow each yields a graph populated under the same node and edge schema once the corresponding records are parsed through a thin adapter. As a result, a detector trained on one deployment can be reused across heterogeneous agentic systems without retraining on platform-specific schemas, and the framework can be extended to additional runtimes by supplying a new adapter rather than redesigning the representation.

Trust boundary. We assume that workflow measurements collected within an organizational trust boundary are trustworthy, supported by standard system-security mechanisms on the host infrastructure. When execution is delegated to external services or agents outside this boundary, we assume the availability of remote attestation mechanisms for exchanging *verifiable workflow-level evidence* [1, 2, 33, 47]. The design of secure measurement and attestation infrastructures is orthogonal to our contribution.

2.2 Threat Model

Adversary capabilities and limitations. We consider adversaries operating *within* the trusted measurement boundary. An adversary may exploit vulnerabilities in legitimate agents, inject malicious prompts directly through user-facing channels or indirectly through retrieved documents, tool outputs, or shared artifacts, manipulate intermediate execution context that downstream agents consume, or abuse the legitimate tool-call interface of compromised or coerced agents to steer downstream execution. Crucially, a sophisticated adversary may *launder* malicious intent across multiple agents so that each individual step looks acceptable in isolation [14, 23]. The adversary cannot, however, forge agent identities at the runtime layer, compromise the integrity of workflow measurements, or directly tamper with the detector itself; these guarantees are inherited from standard system-security mechanisms and from remote attestation across organizational boundaries.

Anomalies in scope. Within this trust boundary, we consider two classes of anomalies that both manifest as abnormal workflow execution. *Adversarial workflow violations* arise from intentional manipulations such as prompt injection, tool-description injection, privilege escalation, tool misuse, or coordinated multi-agent compromise. *Intrinsic workflow failures* arise from the system itself, including hallucinated intermediates, flawed planning, ambiguous instructions, coordination breakdowns, runaway retries, and other forms of execution instability. Although their root causes differ, both yield workflows that diverge from the structural and semantic regularities of benign execution; SKYNET is designed to detect both under a single decision rule. We further assume that a representative corpus of benign executions is available at training time; the corpus need not be exhaustive, and previously unseen anomalies and zero-day attacks need not appear in any form during training.

2.3 Problem Definition

Consider an agentic AI system of the form described in Sec. 2.1. For every user request, the system produces an execution that is recorded, through the workflow construction of Sec. 2.1, as a directed attribute graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, where $\mathcal{V}$ is the set of observed execution steps (planning actions, tool invocations, inter-agent messages, system events), $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of directed dependency edges whose direction encodes both temporal ordering and causal precedence (temporal sequencing, delegation, data flow, and cross-agent communication), and $\mathcal{X}$ collects the node- and edge-level attributes that summarize each step’s actor role, action category, invoked tool/endpoint, and execution status, together with edge interaction semantics. Let $\mathcal{G}_{\text{all}}$ denote the space of all such graphs producible by the system. Each executed workflow is modeled as a sample from one of two unknown distributions over $\mathcal{G}_{\text{all}}$: a benign distribution $\mathbb{P}_b$ governing workflows that realize intended task behavior, and an anomalous distribution $\mathbb{P}_a$ governing workflows affected by the adversarial manipulations and intrinsic failures of Sec. 2.2. At training time, only $\mathbb{P}_b$ is observable, through a finite corpus $\mathcal{D}_b = \{\mathcal{G}_i\}_{i=1}^N \stackrel{\text{iid}}{\sim} \mathbb{P}_b$ of measured benign workflows; no labeled anomalies are available.

Workflow-level anomaly detection. Given $\mathcal{D}_b$ and a false-positive budget $\epsilon \in (0, 1)$, the goal is to learn a graph-level scoring function $S : \mathcal{G}_{\text{all}} \rightarrow \mathbb{R}_{\geq 0}$ and a threshold $\eta \in \mathbb{R}_{\geq 0}$ that induce the decision rule $\delta_\eta(\mathcal{G}) = \mathbf{1}[S(\mathcal{G}) > \eta]$, flagging $\mathcal{G}$ as anomalous when its score exceeds η . We formulate the design problem as

$$\min_{S \in \mathcal{S}, \eta \geq 0} \mathbb{P}_{\mathcal{G} \sim \mathbb{P}_a}[S(\mathcal{G}) \leq \eta] \quad \text{s.t.} \quad \mathbb{P}_{\mathcal{G} \sim \mathbb{P}_b}[S(\mathcal{G}) > \eta] \leq \epsilon, \quad (1)$$

where $\mathcal{S}$ is a hypothesis class of scoring functions trainable from $\mathcal{D}_b$ without anomaly labels. Because $\mathbb{P}_b$ is accessible only through $\mathcal{D}_b$, the false-positive constraint in (1) is enforced empirically by calibrating η as a high quantile of the benign-score distribution $\{S(\mathcal{G}_i)\}_{i=1}^N$ at level $q = 1 - \epsilon$. The unit of decision is the entire workflow $\mathcal{G}$ rather than any individual step or message, so the problem is intrinsically *graph-level*: S must reflect non-local regularities of benign agentic execution that span multiple steps and multiple agents. The instantiation of $\mathcal{S}$, the construction of S , and the calibration of η that realize (1) are detailed in Sec. 3.

3 The SKYNET Framework

3.1 Overview and Design Rationale

SKYNET realizes the workflow-level scoring function S , the threshold η , and the hypothesis class $\mathcal{S}$ sought in problem (1) as an end-to-end pipeline that consumes the workflow graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ defined in Sec. 2.3 and produces a single graph-level decision rule. The pipeline is designed as a *principled* composition rather than a one-off model: every stage is anchored to a property of agentic AI execution, so the detector is benign-only by construction, graph-level by construction, and class-agnostic with respect to whether the underlying anomaly is intrinsic or adversarial.

Training pipeline (Figure 2) consists of four stages applied to the benign workflow corpus $\mathcal{D}_b = \{\mathcal{G}_i\}_{i=1}^N$. *Stage T1: workflow-aware view construction* (Sec. 3.2) expands each benign anchor $\mathcal{G}_i$ into an invariant-preserving positive view $\mathcal{G}_i^+$ and a set of counterfactual invariant-violating negative views $\mathcal{G}_i^-$; both are grown from the

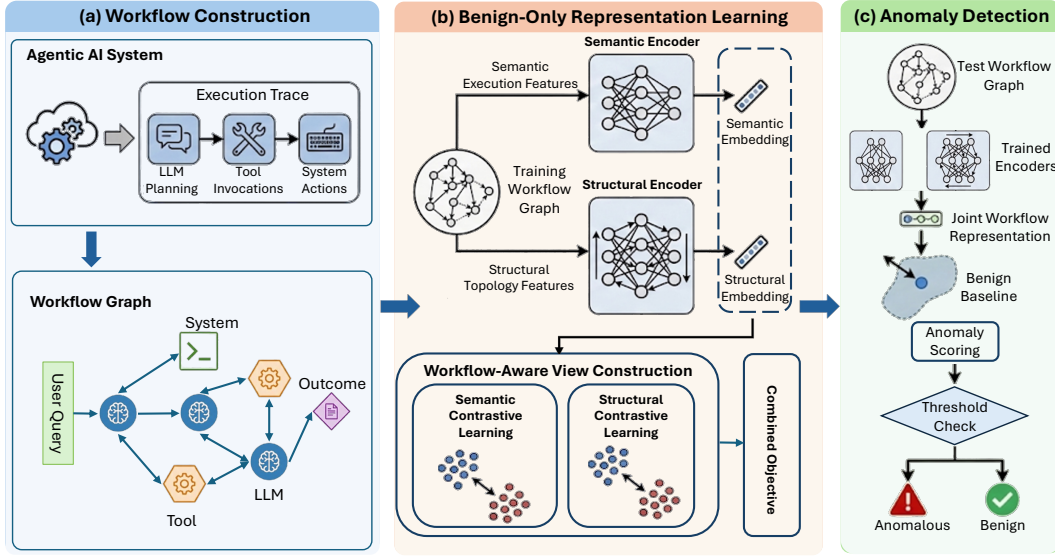


Figure 2: Overview of SKYNET. SKYNET constructs a workflow graph from an observed agentic AI execution, learns benign workflow representations from semantic and structural signals, and detects anomalies by measuring how far a new workflow deviates from learned benign patterns.

structural regularities of benign agentic execution rather than from random graph perturbations, which is what allows the resulting training signal to live near the boundary of benign behavior instead of drifting off into trivially separable noise. *Stage T2: dual-branch encoding* (Sec. 3.3) maps each workflow through a semantic encoder f_θ that summarizes execution context and a structural encoder g_ϕ that summarizes dependency organization, separating *what happens* from *how effects propagate* so that anomalies hiding on either channel remain visible. *Stage T3: workflow-aware contrastive learning* jointly trains f_θ and g_ϕ to pull each anchor toward its invariant-preserving partner and push it away from its counterfactual under a single workflow-level objective derived from agentic workflow invariants. *Stage T4: benign-mode calibration* (Sec. 3.4) consolidates the learned joint embeddings of $\mathcal{D}_b$ into a bank of K benign execution modes $\{c_k\}_{k=1}^K$ and sets the threshold η to the q -quantile of benign training scores with $q = 1 - \epsilon$, exactly matching the false-positive constraint in (1) without exposure to anomaly labels.

Inference pipeline (Figure 2, bottom) takes a new workflow $\mathcal{G}$ through the same graph construction and dual-branch encoding, scores it as $S(\mathcal{G}) = \min_k \|z(\mathcal{G}) - c_k\|_2^2$ in the joint embedding space, and flags it whenever $S(\mathcal{G}) > \eta$. The same single rule covers adversarial manipulations, intrinsic failures, and previously unseen anomalies, since every off-manifold workflow surfaces under the same distance, and the procedure runs in two deployment regimes: a per-workflow trailing monitor that scores completed executions, and an incremental per-step monitor that re-scores in-flight executions on every newly observed step. Algorithm 1 consolidates the full procedure.

Design rationale. Three properties of agentic AI execution justify the choices above. (O1) *Dual-channel manifestation.* Agentic anomalies manifest on two distinct channels: many adversarial

manipulations preserve the surface vocabulary of agent actions while reshaping dependency flow, and many intrinsic failures preserve the dependency skeleton while corrupting the semantics of intermediate steps; a single representation collapses these channels and misses whichever class hides on the channel it underweights, motivating the dual semantic–structural branches in T2. (O2) *Label-free benign manifold.* Anomaly labels are scarce, inconsistent across deployments, and inherit the blind spots of whichever threat catalog produced them, so the training signal is grown from benign workflows alone, with positives and negatives generated from agentic workflow invariants rather than the random graph augmentations of generic graph contrastive learning, which drift off the benign manifold and yield trivially separable negatives. (O3) *Multimodal benign behavior.* Benign agentic behavior is intrinsically heterogeneous (orchestrator pipelines, ReAct loops, deep-research workflows, and multi-agent debates are all benign yet follow qualitatively different normal regimes), so normality is represented as a set of benign execution modes rather than a single reference, preventing legitimate but atypical workflows from being conflated with anomalies.

3.2 Workflow-Aware View Construction

The cornerstone of SKYNET’s benign-only learning is a pair of view-construction operators $\mathcal{T}^+ : \mathcal{G}_{\text{all}} \rightarrow \mathcal{G}_{\text{all}}$ and $\mathcal{T}^- : \mathcal{G}_{\text{all}} \rightarrow 2^{\mathcal{G}_{\text{all}}}$, which, for every benign anchor $\mathcal{G}_i \in \mathcal{D}_b$, produce a positive view $\mathcal{G}_i^+ = \mathcal{T}^+(\mathcal{G}_i)$ and a finite set of counterfactual negative views $\mathcal{G}_i^- = \mathcal{T}^-(\mathcal{G}_i)$. In contrast to the random graph augmentations used in conventional contrastive learning, $\mathcal{T}^+$ and $\mathcal{T}^-$ are parameterized by *workflow invariants*, that is, properties that benign agentic executions consistently exhibit and that known agentic anomaly classes are known to perturb.

Workflow invariants. We identify four such invariants. (i) *Role consistency:* agent responsibilities and delegation patterns remain

compatible with their intended functions. (ii) *Provenance and trust consistency*: outputs consumed by later steps originate from expected sources, and benign workflows do not silently elevate the influence of low-trust artifacts on high-stakes downstream actions. (iii) *Dependency and coordination consistency*: prerequisite relations, hand-off structure, and information flow remain coherent, with delegation closing properly and inter-agent coordination converging. (iv) *Stability*: executions remain bounded and avoid uncontrolled retries, loops, or oscillatory propagation. Each invariant is empirically associated with one or more known anomaly families: prompt injection commonly violates role and provenance consistency [9, 45]; coordination breakdowns and intrinsic planning failures violate dependency consistency [49]; runaway retries and progress-misjudgment failures violate stability [50]. Anchoring view construction on these invariants is what makes SKYNET workflow-aware rather than a generic graph learner, and what gives the resulting training signal a semantic interpretation grounded in the agentic-AI threat surface.

Invariant-preserving views (positives). The positive operator $\mathcal{T}^+$ maps $\mathcal{G}_i$ to a view $\mathcal{G}_i^+$ that respects all four invariants while exposing the natural variability of benign agentic execution. We instantiate $\mathcal{T}^+$ as a stochastic composition of three families of transformations: (i) *observation-preserving* rewrites retain the dependency structure while masking or perturbing non-essential observation content (e.g., paraphrasing free-form notes, redacting irrelevant tool fields); (ii) *execution-equivalent* rewrites apply local edits that preserve task semantics, such as reordering independent operations or substituting semantically equivalent tool/action attributes; and (iii) *prefix or partial* rewrites expose incomplete yet still valid sub-workflows, reflecting the partial observability faced by an in-flight monitor. By construction $\mathcal{G}_i^+$ remains in $\text{supp}(\mathbb{P}_b)$ and is therefore a *semantically faithful* positive: pulling $\mathcal{G}_i$ and $\mathcal{G}_i^+$ together in representation space teaches the encoders to be invariant to the natural variability of benign agentic execution rather than to arbitrary graph noise.

Counterfactual invariant-violating views (negatives). The negative operator $\mathcal{T}^-$ is the design move that distinguishes SKYNET's contrastive scheme from generic graph contrastive learning. For each benign anchor $\mathcal{G}_i$ and invariant $I \in \{\text{ROLE, PROV, DEP, STAB}\}$, an invariant-specific operator $\mathcal{T}_I^-$ produces a counterfactual $\mathcal{G}_i^{-,I}$ that minimally violates I while leaving the remaining three invariants intact. Concretely, role counterfactuals reassign or relabel an agent's action so that a delegation crosses a benign role boundary; provenance counterfactuals reroute one downstream consumption to draw from a non-canonical source; dependency counterfactuals delete or insert a single hand-off edge so that a sub-workflow no longer closes properly; stability counterfactuals extend a retry or loop tail beyond benign envelopes. Each $\mathcal{G}_i^{-,I}$ obeys three principles by design: *minimal modification* (smallest perturbation sufficient to violate the targeted invariant), *single-invariant violation* (the training signal isolates one axis of agentic anomaly), and *local plausibility* (the perturbed workflow could plausibly arise in real agentic execution). The full negative set $\mathcal{G}_i^- = \bigcup_I \{\mathcal{G}_i^{-,I}\}$. Therefore, it carves out the *boundary* of $\text{supp}(\mathbb{P}_b)$ along agentic-meaningful directions, in contrast to random edge or attribute perturbations

whose negatives drift far outside $\text{supp}(\mathbb{P}_b)$ and are trivially separable. The ablation in Sec. 4.3 confirms that replacing $\mathcal{T}^-$ with random graph augmentation collapses recall on near-manifold attacks and intrinsic failures even when the rest of the pipeline is held fixed.

3.3 Dual-Branch Workflow Representation Learning

Two-branch encoder. Given a workflow graph $\mathcal{G}$, SKYNET computes two embeddings, $h(\mathcal{G}) = f_\theta(\mathcal{G}) \in \mathbb{R}^{d_h}$ and $t(\mathcal{G}) = g_\phi(\mathcal{G}) \in \mathbb{R}^{d_t}$, where the *semantic encoder* f_θ is a message-passing graph encoder that consumes node and edge attributes (actor role, action category, tool/endpoint, status, edge type), and the *structural encoder* g_ϕ is a message-passing graph encoder that consumes the structural descriptors of each node (degree statistics, distances to workflow boundaries, ancestor/descendant relations, and influence-concentration scores). The two branches do not share parameters; this decoupling is intentional and reflects observation (O1): agentic anomalies frequently perturb only one of the two channels, and parameter sharing would force a single representation to compromise between them.

Per-anchor contrast pool. Let $\mathcal{B} \subseteq \mathcal{D}_b$ be a mini-batch of size B . For each anchor $\mathcal{G}_i \in \mathcal{B}$, we instantiate $\mathcal{G}_i^+ = \mathcal{T}^+(\mathcal{G}_i)$ and $\mathcal{G}_i^- = \mathcal{T}^-(\mathcal{G}_i)$ on the fly and define the per-anchor contrast pool $\mathcal{Q}_i = \{\mathcal{G}_j^+ : j \in \mathcal{B}, j \neq i\} \cup \mathcal{G}_i^-$, which combines in-batch benign negatives with same-anchor counterfactual negatives. The two sources are complementary: in-batch benign negatives spread different benign workflows across the embedding space and prevent representation collapse, while same-anchor counterfactual negatives sharpen the decision surface along the agentic-invariant directions that downstream anomalies actually perturb.

Dual contrastive objective. Let $\text{sim}(u, v) = u^\top v / (\|u\| \|v\|)$ denote cosine similarity and $\tau > 0$ a temperature. The semantic and structural branches are trained under analogous standard contrastive objectives,

$$L_{\text{sem}}(\theta) = -\frac{1}{B} \sum_{i=1}^B \log \frac{\exp(\text{sim}(h_i, h_i^+)/\tau)}{\sum_{\mathcal{G} \in \{\mathcal{G}_i^+\} \cup \mathcal{Q}_i} \exp(\text{sim}(h_i, h(\mathcal{G}))/\tau)}, \quad (2)$$

$$L_{\text{str}}(\phi) = -\frac{1}{B} \sum_{i=1}^B \log \frac{\exp(\text{sim}(t_i, t_i^+)/\tau)}{\sum_{\mathcal{G} \in \{\mathcal{G}_i^+\} \cup \mathcal{Q}_i} \exp(\text{sim}(t_i, t(\mathcal{G}))/\tau)}, \quad (3)$$

where $h_i = h(\mathcal{G}_i)$, $h_i^+ = h(\mathcal{G}_i^+)$, and t_i, t_i^+ are defined analogously. The two branches are combined into the workflow-level training objective

$$L(\theta, \phi) = \alpha L_{\text{sem}}(\theta) + \beta L_{\text{str}}(\phi), \quad \alpha, \beta > 0, \quad (4)$$

which is minimized by stochastic gradient descent over the two encoders' parameters. Intuitively, L_{sem} teaches f_θ that two workflows with the same task semantics but different surface presentation should embed close, and that a workflow whose role, provenance, dependency, or stability invariant has been minimally violated should embed far; L_{str} enforces the analogous property in the structural channel. Concretely, the gradient of L_{sem} with respect to h_i is a softmax-weighted sum that pulls h_i toward h_i^+ and pushes it away

from each $h(\mathcal{G})$ for $\mathcal{G} \in Q_i$, with the strongest repulsion applied to whichever counterfactual currently sits closest to the anchor in representation space; the same dynamics hold for L_{str} in the structural channel. Through the standard mutual-information lower-bound interpretation of contrastive learning, minimizing (2)–(3) maximizes a tractable lower bound on $I(h(\mathcal{G}); h(\mathcal{G}^+))$ and $I(t(\mathcal{G}); t(\mathcal{G}^+))$, so the encoders retain exactly the information that is invariant under the benign view operator $\mathcal{T}^+$ and discard the information that the counterfactual operator $\mathcal{T}^-$ shows is *not* invariant under benign behavior. The novelty of (4) therefore lies not in the contrastive form itself but in *what it contrasts*: invariant-preserving and invariant-violating views, applied along a dual semantic–structural decomposition that mirrors the two channels through which agentic anomalies manifest.

Hypothesis class. The hypothesis class $\mathcal{S}$ from problem (1) is therefore realized as $\mathcal{S} = \{ \mathcal{G} \mapsto \min_k \|z_{\theta, \phi}(\mathcal{G}) - c_k\|_2^2 \mid (\theta, \phi) \in \Theta, \{c_k\}_{k=1}^K \subset \mathbb{R}^{d_h + d_t} \}$, with $z_{\theta, \phi}(\mathcal{G}) = [h(\mathcal{G}); \sqrt{\lambda} t(\mathcal{G})]$. The encoder parameters (θ, ϕ) are fit by minimizing (4); the mode anchors $\{c_k\}$ and threshold η are fit subsequently on $\mathcal{D}_b$ alone, as described next.

3.4 Workflow-Mode Anomaly Scoring

After training, SKYNET forms the joint workflow representation $z(\mathcal{G}) = [h(\mathcal{G}); \sqrt{\lambda} t(\mathcal{G})] \in \mathbb{R}^{d_h + d_t}$, with $\lambda > 0$ rescaling the structural branch. To respect the heterogeneity of benign agentic execution (O3), normality is modeled as a set of K *benign execution modes* $\{c_k\}_{k=1}^K$ obtained by k -means on $\{z(\mathcal{G}_i)\}_{i=1}^N$, where each mode corresponds to a recurrent pattern of nominal behavior such as a coordination style, a tool-usage regime, or a task-decomposition structure. The graph-level anomaly score S in problem (1) is then instantiated as the squared distance to the nearest benign mode,

$$S(\mathcal{G}) = \min_{1 \leq k \leq K} \|z(\mathcal{G}) - c_k\|_2^2,$$

and the threshold η is calibrated as the empirical q -quantile of $\{S(\mathcal{G}_i)\}_{i=1}^N$ with $q = 1 - \epsilon$, exactly enforcing the false-positive constraint of (1) on $\mathcal{D}_b$. A test workflow is flagged when $S(\mathcal{G}) > \eta$. Because S measures distance to the benign manifold rather than similarity to any specific anomaly family, the same single decision rule covers adversarial manipulations, intrinsic failures, and previously unseen anomalies: any workflow that fits none of the benign modes surfaces as off-manifold geometry under S . The end-to-end procedure is summarized in Algorithm 1.

4 Evaluation

We evaluate SKYNET as an always-on workflow-level anomaly detector around three questions. **RQ1.** Does it detect workflow-level anomalies across both adversarial manipulations and intrinsic failures while sustaining a deployment-relevant low-FPR operating point (Sec. 4.2)? **RQ2.** Does each workflow-aware learning component deliver the operating-point gain it is designed for (Sec. 4.3)? **RQ3.** Is its runtime overhead compatible with always-on monitoring (Sec. 4.4)?

Algorithm 1 SKYNET: benign-only workflow-level anomaly detection.

Require: Benign workflow-graph corpus $\mathcal{D}_b = \{\mathcal{G}_i\}_{i=1}^N$; semantic encoder f_θ ; structural encoder g_ϕ ; loss weights α, β ; structural scale λ ; temperature τ ; benign-mode count K ; calibration quantile q ; training epochs E ; batch size B .

Ensure: Trained encoders (f_θ, g_ϕ) ; benign-mode bank $\{c_k\}_{k=1}^K$; detection threshold η ; anomaly scorer $S(\cdot)$.

- 1: Initialize θ, ϕ .
- 2: **for** epoch = 1 **to** E **do**
- 3: Sample a mini-batch $\mathcal{B} \subset \mathcal{D}_b$ of size B .
- 4: For each $\mathcal{G}_i \in \mathcal{B}$, generate an invariant-preserving view $\mathcal{G}_i^+$ and a set of counterfactual views $\mathcal{G}_i^-$ that each minimally violate one role, provenance, dependency, or stability invariant.
- 5: Form the per-anchor contrast pool $Q_i \leftarrow \{\mathcal{G}_i^+\}_{j \neq i} \cup \mathcal{G}_i^-$.
- 6: Compute h_i, h_i^+, t_i, t_i^+ for $\mathcal{G}_i, \mathcal{G}_i^+$ and $h(\mathcal{G}), t(\mathcal{G})$ for all $\mathcal{G} \in Q_i$.
- 7: $L_{\text{sem}} \leftarrow -\frac{1}{B} \sum_i \log \frac{\exp(\text{sim}(h_i, h_i^+)/\tau)}{\sum_{\mathcal{G} \in \{\mathcal{G}_i^+\} \cup Q_i} \exp(\text{sim}(h_i, h(\mathcal{G}))/\tau)}$.
- 8: $L_{\text{str}} \leftarrow -\frac{1}{B} \sum_i \log \frac{\exp(\text{sim}(t_i, t_i^+)/\tau)}{\sum_{\mathcal{G} \in \{\mathcal{G}_i^+\} \cup Q_i} \exp(\text{sim}(t_i, t(\mathcal{G}))/\tau)}$.
- 9: $L \leftarrow \alpha L_{\text{sem}} + \beta L_{\text{str}}$; update θ, ϕ by one optimizer step on L .
- 10: **end for**
- 11: Compute $\{z(\mathcal{G}_i) = [h(\mathcal{G}_i); \sqrt{\lambda} t(\mathcal{G}_i)]\}_{i=1}^N$ and fit K benign-mode anchors $\{c_k\}_{k=1}^K$ by k -means in the joint embedding space.
- 12: Define $S(\mathcal{G}) \leftarrow \min_k \|z(\mathcal{G}) - c_k\|_2^2$; set η to the q -quantile of $\{S(\mathcal{G}_i)\}_{i=1}^N$.
- 13: **At inference**, given a (possibly in-flight) test workflow prefix $\mathcal{G}$:
- 14: compute $z(\mathcal{G}) = [f_\theta(\mathcal{G}); \sqrt{\lambda} g_\phi(\mathcal{G})]$ and $s \leftarrow S(\mathcal{G})$.
- 15: **if** $s > \eta$ **then** flag $\mathcal{G}$ as anomalous **else** accept as benign.
- 16: The same procedure is applied either once on the completed workflow (trailing per-workflow mode) or after each newly observed step on the current prefix (incremental per-step mode).
- 17: **return** $(f_\theta, g_\phi, \{c_k\}, \eta, S)$.

4.1 Evaluation Setup

Benchmarks. For adversarial evaluation we use Agent-SafetyBench [48] (111 trajectories from safety red-teaming) and ATBench [20] (497 trajectories across eight risk sources and fourteen failure modes); for intrinsic failures we use AgentErrorBench [50] (200 non-adversarial trajectories across AlfWorld, GALA, and WebShop). A *Combined* scenario aggregates all 808 anomalies with the benign test set to measure cross-source robustness when the anomaly distribution broadens. Benign trajectories come from ETO [30] on AlfWorld [29], SciWorld [36], and WebShop [42], converted to workflow graphs (Sec. 3) and partitioned into 4,494/562/562 train/validation/test splits; anomalies are never seen during training. The three anomaly benchmarks are public datasets independently released by different groups; using them together avoids co-design bias between training and evaluation.

Baselines. We compare against six detectors covering the current design space: agent-centric textual monitoring (GUARDAGENT [40]); provenance-level defenses (TRACEAGIS [21], GUARDIAN [49]);

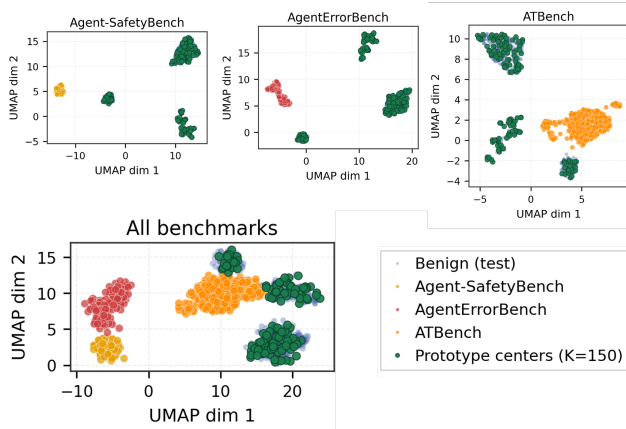


Figure 3: UMAP projection of the learned joint workflow embedding $z(\mathcal{G})$. Top-row panels overlay anomalies from each benchmark on the held-out benign test set; the bottom panel shows all three benchmarks together with the $K=150$ benign mode anchors. Benign workflows form a coherent manifold around the anchors; anomalies from each source consistently lie off-manifold.

and graph anomaly detectors (F-GAE [11], CoLA [22], TopoGCL[6]). All baselines share the same graph inputs and benign training split, each tuning its own threshold via a balance-oriented validation rule. To prevent trivial detection via surface features, graph baselines are stripped of text- and length-based shortcuts: raw text content is removed from node attributes and only categorical/numerical workflow features (role, action type, tool, status, structural descriptors) are retained, and per-graph node/edge counts are normalized so that graph-size cues alone cannot drive the score.

Implementation and metrics. We set structural scale $\lambda = 1.2$ and combine the semantic and structural objectives with $\alpha = 0.6, \beta = 0.4$. Normality is modeled by a benign-only k -means bank with $K = 150$ modes, and the threshold is calibrated on benign training scores at a quantile that targets a sub-1% benign false-positive budget, yielding a single fixed operating point that materializes to $\approx 0.89\%$ FPR on the held-out benign test split. We report accuracy, precision, recall, F1, and FPR; FPR is the deployment-critical metric, since any always-on monitor firing above a few percent FPR is unusable in production. ROC curves provide a threshold-independent view that controls for the asymmetry between SKYNET’s fixed-quantile calibration and the per-method balance-oriented thresholds used by the baselines.

4.2 Detection Effectiveness (RQ1)

Embedding-space geometry. Figure 3 qualitatively illustrates how SKYNET reshapes the workflow space: benign workflows concentrate in a coherent manifold around the mode anchors, while anomalies from every benchmark lie off-manifold regardless of which threat class produced them. This is the geometry the multi-mode detector exploits. UMAP is nonlinear and stochastic, so the 2D view is illustrative; all detection scores below are computed in the original joint embedding.

Cross-scenario detection. Table 1 and Figure 4 show that SKYNET is the only detector that maintains high recall with sub-1%

FPR across all four scenarios at one fixed operating point, as required for online monitoring. It achieves 100% recall at 0.89% FPR on Agent-SafetyBench and AgentErrorBench, delivers the best recall-FPR trade-off on ATBench and Combined, and attains the highest F1 on both broader scenarios. The Combined setting most clearly distinguishes workflow-level monitoring from prior approaches. On the focused benchmarks, several baselines also achieve near-saturated recall below 1% FPR (e.g., GUARDIAN matches SKYNET’s recall at lower FPR), as coarse trace or coordination shifts are easy for graph monitors to detect. Under broader threat distributions, however, they degrade sharply: GUARDIAN’s recall falls to the tens of percent on ATBench and roughly halves on Combined, while TRACEAEGIS loses most recall on ATBench, indicating poor transfer of coordination- or trace-specific signals. In contrast, SKYNET detects off-manifold deviations from benign workflow regularities, allowing it to transfer across heterogeneous adversarial and intrinsic anomalies while preserving both high recall and the sub-1% FPR budget, also reflected in the highest mean F1. The ROC curves confirm this threshold-independently: text-only monitoring is insufficient at workflow scale; trace- and provenance-level defenses perform well on focused sets but degrade under distribution shift; and generic graph anomaly detectors require 2–6% FPR for competitive recall, making them unsuitable for always-on monitoring.

Generality across agentic anomaly regimes. Table 2 breaks down failure types and adversarial behaviors under the same fixed detector. On AgentErrorBench, SKYNET achieves full recall across all 17 intrinsic failure types, including subtle errors such as hallucination, constraint ignorance, and causal misattribution, indicating a broadly discriminative boundary over failures unseen during training. Lower precision on rare categories is largely a sample-size effect: at the fixed 0.89% FPR, the ~ 5 benign false positives among 562 samples dominate precision when only a few positives are present. On ATBench, recall remains high for overt attacks (e.g., *malicious_tool_execution* and *direct_prompt_injection*) but drops for *inherent_agent_failures* and *indirect_prompt_injection*, whose executions most closely resemble benign workflows. These cases mark the limit of workflow-level off-manifold detection and motivate complementary content-aware checks.

4.3 Workflow-Aware Learning Scheme: Component Contributions (RQ2)

Table 3 quantifies each component’s contribution on Combined using the same data as Table 1, but from a separate ablation run. We prioritize recall and FPR, using F1 only as a secondary check, since even modest recall loss or a few-percent FPR increase is costly for always-on monitoring. **Structural branch.** Removing it reduces recall by several points and raises FPR to $\sim 0.9\%$, with a corresponding F1 drop, showing that dependency-sensitive structure adds information beyond semantics; many agentic failures arise from how effects propagate through a workflow rather than from local step content. **Counterfactual invariant-violating views.** Removing them increases FPR by roughly 1.5 $\times$, exceeding the 1% budget, while only modestly reducing recall, indicating that they primarily tighten the benign boundary against near-manifold deviations. **Multi-mode benign scoring.** Collapsing the $K=150$ benign modes into a single reference is the most damaging ablation: recall

Method	Agent-SafetyBench					AgentErrorBench					ATBench					Combined				
	Acc	Prec	Rec	F1	FPR	Acc	Prec	Rec	F1	FPR	Acc	Prec	Rec	F1	FPR	Acc	Prec	Rec	F1	FPR
SKYNET	99.26	95.69	100.00	97.80	0.89	99.34	97.56	100.00	98.77	0.89	91.41	98.80	82.70	90.03	0.89	93.36	99.31	89.36	94.07	0.89
F-GAE	97.86	90.91	99.10	94.83	2.14	98.05	93.02	98.50	95.68	2.14	88.94	90.12	84.31	87.12	2.14	90.87	93.44	86.92	90.06	2.14
TRACEAEGIS	96.62	86.89	96.40	91.40	3.38	96.83	89.19	94.50	91.77	3.38	67.43	61.54	34.61	44.27	3.38	75.92	78.48	58.42	66.98	3.38
GUARDIAN	99.85	99.11	100.00	99.55	0.18	99.87	99.50	100.00	99.75	0.18	61.57	98.91	18.31	30.90	0.18	70.29	99.75	49.75	66.39	0.18
TOPOGCL	95.69	80.15	98.20	88.26	4.80	92.78	86.43	86.00	86.22	4.80	56.56	70.33	12.88	21.77	4.80	64.23	92.74	42.70	58.47	4.80
CoLA	95.10	77.08	100.00	87.06	5.87	71.26	29.79	7.00	11.34	5.87	73.56	88.34	50.30	64.10	5.87	65.99	91.91	46.41	61.68	5.87
GUARDAGENT	84.10	64.29	8.11	14.40	0.89	73.10	0.00	0.00	0.00	0.89	60.25	94.19	16.30	27.79	0.89	47.23	94.74	11.14	19.93	0.89

Table 1: Detection performance across the four evaluation scenarios (all values in %).

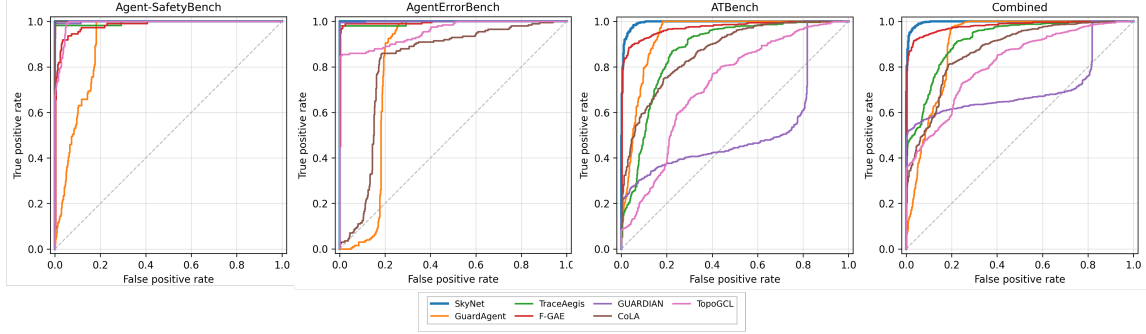


Figure 4: ROC curves of SKYNET and the six baselines across the four evaluation scenarios. SKYNET dominates the deployment-relevant low-FPR region across all four scenarios; baselines that are near-perfect on Agent-SafetyBench and AgentErrorBench degrade sharply on ATBench and Combined.

drops by roughly one third while FPR remains similar, highlighting both the multimodality of benign workflows and the limitation of single-reference one-class scoring. *Workflow-aware view construction*. Removing it lowers recall and raises FPR by about 1.4× beyond the 1% budget, confirming that physically realizable workflow views improve both detection and calibration. Overall, the components are complementary: structural encoding and counterfactual views shape off-manifold geometry, multi-mode scoring captures benign diversity, and workflow-aware views preserve the low-FPR regime required for always-on monitoring.

4.4 Always-On Monitoring Overhead (RQ3)

Table 4 characterizes deployment overhead. Offline per-workflow latency, dominated by graph construction and dual-encoder inference, remains well below one second even for the largest workflows, while mode scoring is negligible. Online per-step latency stays in the tens of milliseconds at early and late prefixes (mean 4.8–48.8 ms at steps 1 and 80), rising to roughly 0.1 s mean (~ 0.57 s P95) at intermediate steps as graph reconstruction grows, before amortizing downward. Thus, SKYNET supports both trailing per-workflow and interactive per-step monitoring without stalling the agent pipeline, meeting the runtime requirements for always-on monitoring at the evaluated scale.

5 Related Work

Agentic AI security. Existing safeguards for agentic AI fall into three broad categories. System-level isolation approaches such as IsolateGPT [39] and ACE [17] decouple execution environments to limit the blast radius of compromised agents. Policy- and analysis-based

defenses such as AgentArmor [35], SafeFlow [19], and ShieldAgent [7] enforce access controls or verify agent actions against predefined safety rules. Runtime monitoring systems such as TraceAegis [21], DRIFT [18], GUARDIAN [49], and MindGuard [37] analyze execution traces or interaction graphs to flag specific threat or coordination patterns. These mechanisms provide important protections at well-defined layers of the agentic stack, but each typically targets a specific violation type or relies on hand-crafted rules; SKYNET is complementary, providing a benign-only, attack-agnostic detection layer that surfaces both adversarial manipulations and inherent failures from observed workflows without relying on attack signatures or behavioral specifications.

Graph-based anomaly detection. Graph anomaly detection has been studied extensively in complex systems. Reconstruction-based methods such as F-GAE [11] flag structural deviations through autoencoder reconstruction error. Contrastive learning approaches such as CoLA [22], AD-GCL [31], and TopoGCL [6] learn anomaly-sensitive graph representations from unlabeled data, while system-log defenses such as MAGIC [15], SLOT [26], and TRAIL [10] apply graph learning to advanced persistent threat detection. These methods provide strong generic graph signal but are not tailored to agentic execution semantics; SKYNET adapts the contrastive paradigm with workflow-aware invariants and counterfactual hard negatives, and pairs it with multi-prototype inference that respects the multimodal nature of benign agentic workflows.

LLM safety and benchmarks. LLM safety research has focused largely on prompt-level vulnerabilities and alignment, including prompt injection attacks [13, 45] and defenses based on filtering or robust training [8, 16]. Benchmarks such as Agent Security

Subclass	<i>n</i>	Acc	Prec	Rec	F1	FPR
AgentErrorBench (failure type)						
inefficient_plan	46	99.18	90.20	100.00	94.85	0.89
over_simplification	20	99.14	80.00	100.00	88.89	0.89
progress_misjudge	20	99.14	80.00	100.00	88.89	0.89
constraint_ignorance	13	99.13	72.22	100.00	83.87	0.89
hallucination	13	99.13	72.22	100.00	83.87	0.89
outcome_misinterpretation	12	99.13	70.59	100.00	82.76	0.89
impossible_action	10	99.13	66.67	100.00	80.00	0.89
step_limit	8	99.12	61.54	100.00	76.19	0.89
environment_error	7	99.12	58.33	100.00	73.68	0.89
misalignment	7	99.12	58.33	100.00	73.68	0.89
causal_misattribution	5	99.12	50.00	100.00	66.67	0.89
parameter_error	4	99.12	44.44	100.00	61.54	0.89
tool_execution_error	4	99.12	44.44	100.00	61.54	0.89
memory_retrieval_failure	2	99.11	28.57	100.00	44.44	0.89
invalid_action	1	99.11	16.67	100.00	28.57	0.89
llm_limit	1	99.11	16.67	100.00	28.57	0.89
unknown	27	99.15	84.38	100.00	91.53	0.89
ATBench (risk source)						
direct_prompt_injection	39	98.84	88.10	94.87	91.36	0.89
indirect_prompt_injection	77	96.09	91.94	74.03	82.01	0.89
tool_description_injection	52	98.37	90.38	90.38	90.38	0.89
jailbreak	61	97.75	91.23	85.25	88.14	0.89
unreliable_or_misinformation	64	97.28	91.23	81.25	85.95	0.89
corrupted_tool_feedback	44	98.35	88.64	88.64	88.64	0.89
malicious_tool_execution	25	99.15	83.33	100.00	90.91	0.89
inherent_agent_failures	135	94.55	95.33	75.56	84.30	0.89

Table 2: Fine-grained detection performance of SKYNET on AgentErrorBench failure types and ATBench risk sources (all values in %).

Variant	Acc	Prec	Rec	F1	FPR
Full SKYNET	96.64	99.48	94.80	97.08	0.71
w/o structural branch	94.09	99.32	90.59	94.76	0.89
w/o counterfactual views	94.09	99.19	90.72	94.76	1.07
w/o workflow-aware aug.	95.84	99.30	93.54	96.33	0.98
single-mode inference	79.42	99.25	65.59	78.99	0.71

Table 3: Component-level ablation on the Combined scenario, with one component disabled at a time (all values in %).

Bench [46], AgentHarm [3], Agent-SafetyBench [48], AgentErrorBench [50], and ATBench [20] provide standardized evaluations for diverse agent vulnerabilities and failure modes. We use the latter three as external testbeds for SKYNET, spanning explicit adversarial manipulations, non-adversarial execution failures, and broad safety-in-the-wild settings.

6 Conclusion

We presented SKYNET, a workflow-level anomaly detection framework for agentic AI systems. Integrated into the runtime observation layer, SKYNET consolidates multi-agent execution traces into directed workflow graphs capturing semantic context and structural dependencies. Trained only on benign workflows, it models normality as a multi-mode benign manifold and detects adversarial manipulations, intrinsic failures, and zero-day anomalies under a unified decision rule. Across three public agentic benchmarks,

Component	Total (s)	Mean (s)	P95 (s)
<i>Per-workflow component cost (offline)</i>			
Graph construction	0.2249	0.0225	0.0882
Encoder forward (sem. + struct.)	0.5816	0.0582	0.1437
Mode scoring (batched)	0.0013	0.0001	—
End-to-end per workflow	—	0.0808	0.2282
<i>End-to-end scalability by workflow size</i>			
Small (12 nodes)	—	0.0243	0.0540
Medium (21–27 nodes)	—	0.0599	0.0681
Large (30–121 nodes)	—	0.1388	0.3114
<i>Online incremental scoring (per-step latency)</i>			
Step 1	—	0.0048	0.0079
Step 10	—	0.0302	0.1237
Step 20	—	0.1048	0.5727
Step 80	—	0.0488	0.0769

Table 4: Runtime overhead of SKYNET: offline per-workflow cost, scalability with workflow size, and online incremental per-step latency.

SKYNET sustains high recall at a sub-1% false positive rate, with per-workflow and per-step latencies suitable for online monitoring.

Acknowledgements

This work was supported in part by the Office of Naval Research under grant N00014-24-1-2663, N00014-24-1-2730, the National Science Foundation under grants 2154930, 2312447, 2247560, 2332675, 2433904, and 2235232, and the Virginia Commonwealth Cyber Initiative (CCI).

References

- [1] Mahmoud Ammar, Ahmed Abdelraoof, and Silviu Vlasceanu. 2024. On bridging the gap between control flow integrity and attestation schemes. In *33rd USENIX Security Symposium (USENIX Security 24)*. 6633–6650.
- [2] Mahmoud Ammar, Adam Caulfield, and Ivan De Oliveira Nunes. 2025. SoK: Integrity, Attestation, and Auditing of Program Execution. In *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 3255–3272. doi:10.1109/SP61157.2025.00077
- [3] Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Dueñas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, Zico Kolter, Matt Fredrikson, Yarin Gal, and Xander Davies. 2025. AgentHarm: A Benchmark for Measuring Harmfulness of LLM Agents. In *International Conference on Learning Representations*. https://proceedings.iclr.cc/paper_files/paper/2025/hash/c493d23af93118975cdbe32cbe7323f5-Abstract-Conference.html
- [4] Anthropic. 2024. Building Effective Agents. <https://www.anthropic.com/engineering/building-effective-agents> Published December 19, 2024.
- [5] Anthropic. 2024. Introducing the Model Context Protocol (MCP). Available at: <https://www.anthropic.com/news/model-context-protocol> (accessed 2026-04-11).
- [6] Yuzhou Chen, Jose Frias, and Yulia R Gel. 2024. TopoGCL: Topological graph contrastive learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 38. 11453–11461.
- [7] Zhaorun Chen, Mintong Kang, and Bo Li. 2025. ShieldAgent: Shielding Agents via Verifiable Safety Policy Reasoning. In *Forty-second International Conference on Machine Learning (ICML)*.
- [8] Edoardo DeBenedetti, Ilia Shumailov, Tianqi Fan, Jamie Hayes, Nicholas Carlini, Daniel Fabian, Christoph Kern, Chongyang Shi, Andreas Terzis, and Florian Tramèr. 2025. Defeating prompt injections by design. *arXiv preprint arXiv:2503.18813* (2025).
- [9] Edoardo DeBenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. 2024. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. *Advances in Neural Information Processing Systems* 37 (2024), 82895–82920.
- [10] Darshan Deshpande, Varun Gangal, Hersh Mehta, Jitin Krishnan, Anand Kannappan, and Rebecca Qian. 2025. TRAIL: Trace Reasoning and Agentic Issue Localization. *arXiv preprint arXiv:2505.08638* (2025).
- [11] Yong Feng, Jinglong Chen, Zijun Liu, Haixin Lv, and Jun Wang. 2022. Full graph autoencoder for one-class group anomaly detection of IIoT system. *IEEE Internet*

- of *Things Journal* 9, 21 (2022), 21886–21898.
- [12] Google Developers Blog. 2025. Announcing the Agent2Agent Protocol (A2A). <https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/>. Accessed 2026-01.
 - [13] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM workshop on artificial intelligence and security*. 79–90.
 - [14] Pengfei He, Yiping Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. 2025. Red-teaming llm multi-agent systems via communication attacks. In *Findings of the Association for Computational Linguistics: ACL 2025*. 6726–6747.
 - [15] Zian Jia, Yun Xiong, Yuhong Nan, Yao Zhang, Jinjing Zhao, and Mi Wen. 2024. MAGIC: Detecting Advanced Persistent Threats via Masked Graph Representation Learning. In *33rd USENIX Security Symposium (USENIX Security 24)*. 5197–5214.
 - [16] Heng Jin, Chaoyu Zhang, Hexuan Yu, Shanghao Shi, Ning Zhang, Y. Thomas Hou, and Wenjing Lou. 2026. Trusting What You Cannot See: Auditable Fine-Tuning and Inference for Proprietary AI. In *35th USENIX Security Symposium (USENIX Security 26)*. USENIX Association, Baltimore, MD.
 - [17] Evan Li, Tushin Mallick, Evan Rose, William Robertson, Alina Oprea, and Cristina Nita-Rotaru. 2026. ACE: A Security Architecture for LLM-Integrated App Systems. In *Network and Distributed System Security Symposium (NDSS)*. doi:10.14722/ndss.2026.230352
 - [18] Hao Li, Xiaogeng Liu, Hung-Chun Chiu, Dianqi Li, Ning Zhang, and Chaowei Xiao. 2025. DRIFT: Dynamic Rule-Based Defense with Injection Isolation for Securing LLM Agents. *Advances in Neural Information Processing Systems* (2025).
 - [19] Peiran Li, Xinkai Zou, Zhuohang Wu, Ruifeng Li, Shuo Xing, Hanwen Zheng, Zhikai Hu, Yiping Wang, Haoxi Li, Qin Yuan, et al. 2025. Safeflow: A principled protocol for trustworthy and transactional autonomous agent systems. *arXiv preprint arXiv:2506.07564* (2025).
 - [20] Yu Li, Haoyu Luo, Yuejin Xie, Yuqian Fu, Zhonghao Yang, Shuai Shao, Qihan Ren, Wanying Qu, Yanwei Fu, Yujiu Yang, et al. 2026. ATBench: A Diverse and Realistic Agent Trajectory Benchmark for Safety Evaluation and Diagnosis. *arXiv preprint arXiv:2604.02022* (2026). doi:10.48550/arXiv.2604.02022
 - [21] Jiahao Liu, Bonan Ruan, Xianglin Yang, Zhiwei Lin, Yan Liu, Yang Wang, Tao Wei, and Zhenkai Liang. 2025. TraceAegis: Securing LLM-Based Agents via Hierarchical and Behavioral Anomaly Detection. *arXiv preprint arXiv:2510.11203* (2025).
 - [22] Yixin Liu, Zhao Li, Shirui Pan, Chen Gong, Chuan Zhou, and George Karypis. 2022. Anomaly detection on attributed networks via contrastive self-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems* 33, 6 (2022), 2378–2392.
 - [23] Xingjun Ma, Yifeng Gao, Yixu Wang, Ruofan Wang, Xin Wang, Ye Sun, Yifan Ding, Hengyuan Xu, Yunhao Chen, Yunhan Zhao, et al. 2025. Safety at Scale: A Comprehensive Survey of Large Model and Agent Safety. *Foundations and Trends in Privacy and Security* 8, 3–4 (2025), 254–469. doi:10.1561/33000000051
 - [24] OpenAI. 2025. Introducing ChatGPT Agent. <https://openai.com/index/introducing-chatgpt-agent/>. Technical blog describing OpenAI's agentic extension of ChatGPT with autonomous planning and tool use.
 - [25] OpenAI. 2025. Model Context Protocol – OpenAI Agents Python Documentation. Available at: <https://openai.github.io/openai-agents-python/mcp/> (accessed 2026-04-11).
 - [26] Wei Qiao, Yebo Feng, Teng Li, Zhuo Ma, Yulong Shen, Jianfeng Ma, and Yang Liu. 2025. Slot: Provenance-driven APT detection through graph reinforcement learning. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*. 963–977.
 - [27] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems* 36 (2023), 68539–68551.
 - [28] Shanghao Shi, Xiao Wang, Chaoyu Zhang, Hao Li, Wenjing Lou, Y. Thomas Hou, Yevgeniy Vorobeychik, Chongjie Zhang, and Ning Zhang. 2026. Think Twice Before You Act: Protecting LLM Agents Against Tool Description Poisoning via Isolated Planning. In *Proceedings of the 43rd International Conference on Machine Learning*.
 - [29] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *International Conference on Learning Representations (ICLR)*.
 - [30] Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and Error: Exploration-Based Trajectory Optimization for LLM Agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*.
 - [31] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. 2021. Adversarial graph augmentation to improve graph contrastive learning. *Advances in Neural Information Processing Systems* 34 (2021), 15920–15933.
 - [32] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024. Voyager: An Open-Ended Embodied Agent with Large Language Models. *Transactions on Machine Learning Research* (2024). <https://openreview.net/forum?id=ehfRiF0R3a>
 - [33] Jinwen Wang, Yujie Wang, Ao Li, Yang Xiao, Ruide Zhang, Wenjing Lou, Y Thomas Hou, and Ning Zhang. 2023. ARI: Attestation of Real-Time Mission Execution Integrity. In *32nd USENIX Security Symposium (USENIX Security 23)*. 2761–2778.
 - [34] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science* 18, 6 (2024), 186345.
 - [35] Peiran Wang, Yang Liu, Yunfei Lu, Yifeng Cai, Hongbo Chen, Qingyou Yang, Jie Zhang, Jue Hong, and Ye Wu. 2025. Agentarmor: Enforcing program analysis on agent runtime trace to defend against prompt injection. *arXiv preprint arXiv:2508.01249* (2025).
 - [36] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. ScienceWorld: Is your Agent Smarter than a 5th Grader?. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11279–11298. doi:10.18653/v1/2022.emnlp-main.775
 - [37] Zhiqiang Wang, Haohua Du, Guanquan Shi, Junyang Zhang, HaoRan Cheng, Yunhao Yao, Kaiwen Guo, and Xiang-Yang Li. 2025. MindGuard: Intrinsic Decision Inspection for Securing LLM Agents Against Metadata Poisoning. *arXiv preprint arXiv:2508.20412* (2025). doi:10.48550/arXiv.2508.20412
 - [38] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. 2024. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*.
 - [39] Yuhao Yu, Franziska Roesner, Tadayoshi Kohno, Ning Zhang, and Umar Iqbal. 2025. IsolateGPT: An Execution Isolation Architecture for LLM-Based Agentic Systems. In *NDSS*.
 - [40] Zhen Xiang, Linzhi Zheng, Yanjie Li, Junyuan Hong, Qinbin Li, Han Xie, Jiawei Zhang, Zidi Xiong, Chulin Xie, Carl Yang, Dawn Song, and Bo Li. 2025. GuardAgent: Safeguard LLM Agents via Knowledge-Enabled Reasoning. In *Proceedings of the 42nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 267)*. PMLR, 68316–68342. <https://proceedings.mlr.press/v267/xiang25a.html>
 - [41] John Zhu, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems* 37 (2024), 50528–50652.
 - [42] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems* 35 (2022), 20744–20757.
 - [43] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=WE_vluYUL-X
 - [44] Hexuan Yu, Chaoyu Zhang, Heng Jin, Shanghao Shi, Ning Zhang, Y. Thomas Hou, and Wenjing Lou. 2026. MINIM: Privacy-Aware Minimal View for Agents via Trusted Local Sanitization. In *Proceedings of the 43rd International Conference on Machine Learning*.
 - [45] Qiusi Zhang, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. INJECAGENT: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents. In *Findings of the Annual Meeting of the Association for Computational Linguistics (ACL) 2024*.
 - [46] Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. 2025. Agent security bench (asb): Formalizing and benchmarking attacks and defenses in llm-based agents. *Proceedings of ICLR 2025* (2025). Comprehensive security evaluation framework including many attack/defense scenarios.
 - [47] Yumei Zhang, Xinzhi Liu, Cong Sun, Dongrui Zeng, Gang Tan, Xiao Kan, and Siqi Ma. 2021. Recfa: Resilient control-flow attestation. In *Proceedings of the 37th Annual Computer Security Applications Conference*. 311–322.
 - [48] Zhixin Zhang, Shiyao Cui, Yida Lu, Jingzhou Zhou, Junxiao Yang, Hongning Wang, and Minlie Huang. 2024. Agent-safetybench: Evaluating the safety of llm agents. *arXiv preprint arXiv:2412.14470* (2024).
 - [49] Jialong Zhou, Lichao Wang, and Xiao Yang. 2025. GUARDIAN: Safeguarding LLM Multi-Agent Collaborations with Temporal Graph Modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*.
 - [50] Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, et al. 2025. Where llm agents fail and how they can learn from failures. *arXiv preprint arXiv:2509.25370* (2025).