



Original Research Paper

Behavioural Fingerprinting of Autonomous AI Agents from Tool-Call Sequences

S Vimal¹, M. Kaliappan², Gaurav Dhiman^{3,4,5}, S. Raja Mohamed²

¹Department of Computer Science and Engineering, KIT- Kalaingar Karunanidhi Institute of Technology, Coimbatore, Tamil Nadu 641402 India

²Department of Artificial Intelligence and Data Science, Ramco Institute of Technology, Rajapalayam, Tamil Nadu 626117, India

³Department of Computer Science and Engineering, Yuan Ze University, Tao Yuan, Taiwan

⁴University Centre for Research and Development, Chandigarh University, Mohali, India

⁵Division of Research and Development, Lovely Professional University, Phagwara, India

Email: kaliappan@ritrjpm.ac.in, svimal@kitcbe.ac.in, gdhiman0001@gmail.com, rajadelip@gmail.com

Abstract

Autonomous agents built on large language models (LLMs) increasingly act in the world by emitting sequences of tool calls, yet the operators who deploy, audit, and secure these agents have few lightweight means of characterising what an agent is doing from its externally observable behaviour alone. This paper investigates whether the operational context of an autonomous agent can be inferred purely from the dynamics of its tool-call trajectory—the length, ordering, repetition, success profile and parameter complexity of its actions—without inspecting which specific tools are invoked or the semantic content of their arguments. We formalise a trajectory as a sequence of typed actions and define a family of feature representations that range from pure structural dynamics to full content-aware descriptors. Using TRAJECT-Bench, a public benchmark of 1670 executable LLM-agent tool-use trajectories spanning nine task domains, we train and cross-validate four classifiers and quantify a clear information gradient. A representation using only sixteen content-agnostic dynamics features attains a macro-F1 of 0.968 (accuracy 0.967, Cohen's κ 0.963, MCC 0.963) against a nine-class random baseline of 0.111, and augmenting these dynamics with generic action-verb n-grams raises macro-F1 to 0.995. All pairwise gains are statistically significant (paired t-test, $p < 0.01$). An early-detection analysis shows that four to five observed actions are sufficient to exceed 90% accuracy, and the dynamics-only model is inexpensive to train and serve (0.16 ms per trajectory at inference). These results provide evidence that the behavioural shape of an agent's actions is itself a discriminative fingerprint, with direct applications to agent observability, anomaly detection, and provenance. We release a fully reproducible single-file benchmark and discuss the path from task-domain fingerprinting toward backbone-model attribution.

Keywords: autonomous LLM agents; tool-call trajectories; behavioural fingerprinting; sequence classification; agent observability; AI security; provenance

*Corresponding Author:

Received: 25th July, 2026 | Revised: 26th August, 2026 | Accepted: 18th September, 2026 | Available Online: 24th September, 2026

(DOI): 10.70102/AEJ.2026.18.8s.7

1. Introduction

Large language models (LLMs) have rapidly evolved from passive text generators into autonomous agents that plan, reason, and act upon external environments through structured tool calls [1, 2, 3]. Modern agent frameworks orchestrate application programming interfaces (APIs), web services, databases, and code interpreters, allowing a model to decompose a natural-language goal into a multi-step trajectory of concrete actions [4, 5, 6, 7]. As these systems move from research demonstrations into production deployments—customer-support automation, financial analysis, travel planning, and software engineering—the trajectory of tool calls an agent emits becomes the primary externally observable record of its behaviour.

This shift creates an acute operational problem. When many heterogeneous agents run concurrently behind a shared gateway or tool-serving layer, the operator responsible for reliability, cost control, and security often observes only a stream of tool invocations with limited or no access to the agent’s prompt, its backbone model, or the semantic content of its intermediate reasoning. Security studies have repeatedly shown that agentic tool use dramatically enlarges the attack surface: indirect prompt injection can hijack an agent into executing attacker-controlled actions [32, 33, 37], and comprehensive threat surveys catalogue dozens of distinct manipulation, exfiltration, and protocol-level exploits unique to LLM agents [34, 35, 36]. In such settings, the ability to characterise “what kind of behaviour is this?” directly from the observable action stream quickly, cheaply, and without privileged access is a prerequisite for monitoring, anomaly detection, and incident response.

A natural question follows: how much can be inferred about an agent from the shape of its actions alone? Intuitively, different operational contexts induce different behavioural rhythms. An agent performing weather look-ups issues short, low-branching sequences with few parameters; an agent conducting financial analysis chains dependent calls with heavy parameterisation and occasional retries; a gaming assistant exhibits yet another cadence. If these rhythms are consistent and separable, then the dynamics of a trajectory, its length, the entropy of its action distribution, its repetition and retry structure, its success profile, and its parameter load—constitute a behavioural fingerprint, analogous to how the sequence of system calls emitted by a process fingerprints the program that produced it in host-based intrusion detection [24, 25, 29].

The central issue this work addresses is that existing tools for reasoning about agent behaviour largely sidestep this observable signal. Trajectory-aware

benchmarks evaluate whether an agent selected the correct tools and produced the correct answer [8, 9, 10, 11, 12, 13], but they measure task success, not the identifiability of the behaviour itself. Conversely, the emerging literature on LLM fingerprinting and attribution establishes that models leave identifiable traces, yet it operates almost exclusively on generated text, weights, or output distributions [14, 15, 16, 17, 18], not on the structural dynamics of tool-use trajectories. To our knowledge, no prior work asks whether the behavioural dynamics of an agent’s tool-call sequence stripped of tool identity and argument contents are sufficient to identify its operational context. This is the gap we fill.

This paper makes the following contributions. (i) We formalise agent behavioural fingerprinting as a sequence-classification problem over typed tool-call trajectories and define a graded family of feature representations that isolates content-agnostic dynamics from content-dependent tool identity. (ii) We construct a rigorous, fully reproducible benchmark on the public TRAJECT-Bench corpus [8], evaluating four classifiers across five feature representations with a complete metric suite (accuracy, macro- and weighted-F1, balanced accuracy, Cohen’s κ , Matthews correlation coefficient, and one-vs-rest ROC-AUC) under stratified cross-validation. (iii) We show that purely structural dynamics sixteen features containing no tool identities whatsoever identify the agent’s task domain with macro-F1 0.968 against a 0.111 chance baseline, and we establish the statistical significance of the information gradient across representations. (iv) We demonstrate practical viability through an early-detection analysis (four to five actions suffice for >90% accuracy) and an efficiency profile suitable for online deployment. (v) We are explicit about scope: the label recovered here is the task domain, not the backbone model, and we delineate precisely why content-based separation is trivial while the dynamics-only result is the scientifically meaningful finding, and how the same pipeline extends to model attribution given appropriately labelled traces.

The remainder of the paper is organised as follows. Section 2 reviews related work across autonomous agents and tool use, trajectory evaluation, LLM fingerprinting, sequence-based intrusion detection, and agent security. Section 3 presents the proposed method, its architecture, and its mathematical formulation. Section 4 describes the experimental setup, and Section 5 reports and interprets the results. Section 6 benchmarks the approach against recent related paradigms. Section 7 discusses limitations, and Section 8 concludes with future directions.

2. Related Work

We organise the literature into five themes that build toward the proposed method: (2.1) autonomous LLM agents and tool use, which define the object of study; (2.2) trajectory-aware evaluation, which established tool-call traces as first-class data; (2.3) fingerprinting and attribution of LLMs, which motivates the identifiability question; (2.4) sequence-based intrusion detection, the closest methodological ancestor; and (2.5) agent security and observability, which supplies the application context.

2.1 Autonomous LLM agents and tool use

The reconceptualisation of LLMs as autonomous agents is surveyed comprehensively by Wang et al. [1] and Xi et al. [2], who describe the now-canonical decomposition of an agent into profiling, memory, planning, and action modules, with tool use as the principal mechanism of action. Qu et al. [3] survey tool learning specifically, taxonomising how models are taught to select, parameterise, and sequence external functions. Foundational techniques include Toolformer [5], which teaches a model to self-annotate API calls; Gorilla [6], which connects an LLM to massive API collections; ToolLLM [7], which scales instruction tuning to over sixteen thousand real-world APIs; and multi-agent orchestration frameworks such as AutoGen [4]. These works establish that tool-call trajectories are the natural behavioural output of modern agents, but they focus on how to produce good trajectories rather than on what an observed trajectory reveals about its producer [42]-[53].

2.2 Trajectory-aware evaluation of agents

A second line of work recognises that evaluating only an agent's final answer is insufficient and that the trajectory itself must be assessed. TRAJECT-Bench [8], the corpus used in this study, pairs executable tools across nine practical domains with tasks grounded in production-style APIs and reports trajectory-level diagnostics including tool selection, argument correctness, and dependency or order satisfaction. τ -bench [9] evaluates agents on multi-step retail and airline tasks; AgentBench [10] spans eight interactive environments; ToolSandbox [11] introduces stateful, conversational tool-use evaluation; and MCP-Bench [12] benchmarks agents on complex real-world tasks served through the Model Context Protocol. Song et al. [13] use trajectories as an optimisation signal for exploration-based agent training. Collectively these benchmarks legitimise the tool-call trajectory as a rich, structured data object; however, their objective is to score task competence, whereas we treat the trajectory as a signal for identifying the behaviour that generated it [54]-[63].

2.3 Fingerprinting and attribution of large language models

The identifiability of machine-learning models is an active research area. LLMmap [15] shows that a model family can be fingerprinted from its responses to a small set of crafted queries, while instructional fingerprinting [16] and HuRef [17] embed or extract human-readable identifiers to protect intellectual property. TRAP [18] performs black-box identification through targeted adversarial prompts, ProFLingo [19] provides a fingerprint-based IP-protection scheme, and scalable and probabilistic schemes extend fingerprinting to many models and to stolen-model hunting [14, 22]. Watermarking [23] embeds detectable statistical signals into generated text. Most directly related to our motivation, Bhardwaj and Mishra [20] identify the underlying LLM inside a deployed generative application, and Bitton et al. [21] detect stylistic fingerprints of LLMs. Crucially, this entire body of work fingerprints models through generated text, weights, or output distributions; none examines the structural dynamics of an agent's tool-call trajectory, which is the modality we study.

2.4 Sequence-based intrusion detection: the closest methodological ancestor

The problem of classifying a process from the sequence of discrete, typed operations it emits is well established in host-based intrusion detection (HIDS), where system-call traces are analysed to distinguish normal from anomalous program behaviour. Sworna et al. [24] systematically review natural-language-processing methods applied to system-call sequences, and Subba and Gupta [25] demonstrate that a TF-IDF vectoriser combined with singular value decomposition detects anomalous system processes, an approach structurally identical to representing a tool-call sequence by weighted n-grams. Kumar and Subba [26] and Mvula et al. [27] apply word-embedding feature extraction to system calls, Wunderlich et al. [29] quantify how the representation of a system call affects detection, and Joraviya et al. [28] render call sequences as images for deep classification. Liao et al. [30] combine dynamic features with a relaxed support-vector machine, and Nakip and Gelenbe [31] propose online self-supervised detection. These works validate the core premise that a sequence of discrete actions carries a recoverable behavioural signature; we transfer this premise from operating-system calls to LLM-agent tool calls, and additionally isolate purely structural dynamics from token identity.

2.5 Agent security, observability, and provenance

The security literature makes the practical case for behavioural characterisation. Greshake et al. [33] introduced indirect prompt injection, Debenedetti et

al. [32] operationalised it against tool-using agents with AgentDojo, and Liu et al. [37] demonstrated injection against LLM-integrated applications. Broad surveys [34, 35] and harmfulness benchmarks [36] document how agency multiplies risk, while Ruan et al. [38] emulate sandboxes to surface agent risks. A recurring theme is the lack of lightweight observability over what agents actually do at the tool layer. Our method contributes precisely such an observable, content-independent signal. Methodologically, our framing also draws on authorship attribution and stylometry [39], where an author is identified from style-bearing but content-independent features, and on classical ensemble learning [40, 41].

3. Proposed Work

We propose a behavioural fingerprinting pipeline that maps an agent's tool-call trajectory to a discriminative representation and classifies its operational context. The design is guided by one principle: to test whether behaviour itself not the identity of the tools invoked carries an identifiable signature, the representation must permit a clean separation between content-agnostic dynamics and content-dependent identity. This section formalises the problem (3.1), defines the feature representations (3.2), specifies the classifier and decision rule (3.3), states the evaluation metrics (3.4), describes the architecture (3.5), and justifies the design choices (3.6).

3.1 Problem formulation

Let an agent execute a task by emitting an ordered trajectory of tool calls. We model a trajectory τ as a finite sequence of typed actions, where each action carries an observed tool (operation) identifier, an execution status, and a set of supplied parameters, as defined in Eq. (1).

$$\tau = (a_1, a_2, \dots, a_T), \quad a_t = (o_t, s_t, \mathbf{I}) \quad (1)$$

Here o_t denotes the invoked tool at step t , $s_t \in \{\text{success}, \text{failure}\}$ its execution status, and $\mathbf{p}_t$ the vector of required and optional parameters supplied to it; $T = |\tau|$ is the trajectory length. Given a labelled corpus of trajectories $\{(\tau_i, y_i)\}$, where $y_i \in C$ is the operational context (here, the task domain), the goal is to learn a classifier $f: \tau \rightarrow C$. A feature map $\phi(\tau)$ transforms the variable-length symbolic trajectory into a fixed-dimensional vector formed by concatenating three complementary views structural, verb, and content as in Eq. (2).

$$\phi(\tau) = [\phi_{\text{struct}}(\tau) \parallel \phi_{\text{verb}}(\tau) \parallel \phi_{\text{content}}(\tau)] \in \iota \quad (2)$$

The design deliberately makes ϕ_{struct} and ϕ_{verb} independent of tool identity, so that the contribution of pure behavioural dynamics can be measured in isolation by ablating ϕ_{content} . This separation is the methodological core of the study and directly enables the information-gradient analysis of Section 5.

3.2 Feature representations

Structural dynamics (ϕ_{struct}). Sixteen scalar features summarise the shape of a trajectory without reference to which tools are used. They comprise length and diversity statistics (number of calls, number of unique tools, unique- and repeat-ratios), the reliability profile (success ratio, failure count and ratio), parameter-complexity statistics (mean and maximum required parameters, their standard deviation, mean and maximum optional parameters, and mean total parameter load), a retry indicator (count of immediately repeated tools), and two entropy terms. The Shannon entropy of the tool-occurrence distribution, Eq. (3), captures how concentrated or exploratory the trajectory is.

$$H(\tau) = - \sum_{o \in \mathcal{O}} \hat{p}(o) \log_2 \hat{p}(o), \quad \hat{p}(o) = \frac{1}{T} \sum_{t=1}^T 1[o_t = o] \quad (3)$$

Repetition and retry structure, Eq. (4), quantify looping and corrective behaviour, where $\text{unique}(\cdot)$ returns the set of distinct tools and $1[\cdot]$ is the indicator function.

$$\rho(\tau) = 1 - \frac{|\text{unique}(o_{1:T})|}{T}, \quad R(\tau) = \sum_{t=2}^T 1[o_t = o_{t-1}] \quad (4)$$

Because ϕ_{struct} contains no tool names, argument values, or textual tokens, any predictive power it exhibits must originate from behavioural dynamics alone. Verb features (ϕ_{verb}) abstract each tool name to a small, fixed vocabulary of sixteen generic action primitives (e.g., search, get, list, create, update, filter) and encode the resulting verb sequence as weighted 1–3-gram term frequencies; these describe the workflow grammar of the trajectory while remaining agnostic to the specific service invoked. Content features (ϕ_{content}) encode the normalised tool-name sequence as TF-IDF-weighted 1–2-grams, Eq. (5), where $f(g, \tau)$ is the frequency of n -gram g in τ , N is the corpus size, and $\text{df}(g)$ is the document frequency of g . This view is intentionally content-dependent and serves as the upper bound of the information gradient.

$$\text{tfidf}(g, \tau) = f(g, \tau) \cdot \log \frac{N}{1 + \text{df}(g)} \quad (5)$$

From these views we construct five nested representations of increasing information: Structural

(ϕ_{struct} only), Verb-ngram (ϕ_{verb} only), Behaviour ($\phi_{\text{verb}} \oplus \phi_{\text{struct}}$, i.e., all content-agnostic signal), Content ($\phi_{\text{content}} \oplus \phi_{\text{struct}}$), and Full (all three). Continuous features are standardised to zero mean and unit variance; sparse n-gram blocks are fused with the dense structural block by horizontal concatenation.

Table 1. Feature representations, from purely behavioural dynamics to full content-aware descriptors.

Representation	Components	Dim.	Uses tool identity?
Structural	16 dynamics scalars	16	No
Verb-ngram	generic action-verb 1–3-grams	≤ 150	No
Behaviour	verb-grams $\oplus$ structural	≤ 166	No
Content	tool-name 1–2-grams $\oplus$ structural	≤ 416	Yes
Full	content $\oplus$ verb $\oplus$ structural	≤ 566	Yes

3.3 Classifier and decision rule

Each representation is classified by four complementary learners: multinomial logistic regression, a random forest [41], extremely randomised trees, and histogram-based gradient boosting, all implemented in scikit-learn [40]. Tree ensembles are chosen because the structural features are expected to interact non-linearly (for example, high retry count is discriminative only in conjunction with low success ratio), which linear models cannot capture. For an ensemble of B estimators, the predicted context is the arg-max of the averaged class posteriors, Eq. (6).

$$\hat{y} = \operatorname{argmax}_{c \in \mathcal{C}} \frac{1}{B} \sum_{b=1}^B P_b(c | \phi(\tau)) \quad (6)$$

3.4 Evaluation metrics

Because the nine classes are only mildly imbalanced, we report a metric suite rather than accuracy alone. The macro-F1 score, Eq. (7), averages per-class F1 and therefore weights every domain equally, penalising models that neglect minority domains.

$$\text{Macro-F1} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \frac{2 P_c R}{P_c + F} \quad (7)$$

We additionally report Cohen’s κ and the multiclass Matthews correlation coefficient (MCC), Eq. (8), both of which correct for chance agreement and are robust under class imbalance; here p_o and p_e are the observed and expected agreements, and the MCC terms are computed from the confusion matrix (c correct predictions, s samples, p_k and t_k the predicted and true totals for class k). We further report balanced accuracy, weighted-F1, and the one-vs-rest macro ROC-AUC.

$$\kappa = \frac{p_o - p_e}{1 - p_e}, \quad \text{MCC} = \frac{c s - \sum_k p_k t_k}{\sqrt{(s^2 - \sum_k p_k^2)(s^2 - \sum_k t_k^2)}} \quad (8)$$

3.5 System architecture

Figure 1 depicts the end-to-end pipeline. A raw tool-call trace is parsed and normalised into the typed action sequence of Eq. (1); three feature extractors then operate in parallel. The dashed region marks the behavior only path structural and verb features which never observes tool identity. Extracted views are fused by $\phi(\tau)$ and passed to the classifier, which emits the predicted operational context. The architecture is modular: the content extractor can be disabled at inference to obtain a strictly content-independent detector, and the same fused representation supports the truncated-trajectory operation used for early detection.

Behavioural Fingerprinting of Autonomous AI Agents — System Pipeline

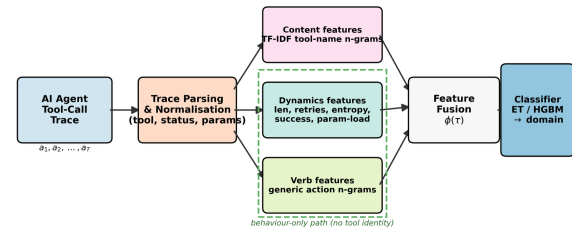


Figure 1. Proposed behavioural-fingerprinting architecture. The dashed region is the content-agnostic path used to isolate the contribution of pure dynamics.

3.6 Design justification and algorithm

Three design decisions distinguish this approach from prior paradigms. First, unlike LLM fingerprinting methods that probe a model with crafted inputs and read its generated text [15, 16, 18], our method is passive and observational: it consumes only the tool-call trace an agent already emits, requiring neither query access nor model internals. Second, unlike trajectory-evaluation benchmarks that score task correctness [8, 10, 11], we treat the trajectory as an identity signal and optimise for discriminability rather than success. Third, and most importantly, the explicit factorisation of $\phi(\tau)$ into content-agnostic and

content-dependent blocks converts a vague question (“can agents be fingerprinted?”) into a measurable gradient, allowing us to attribute predictive power to specific, interpretable behavioural properties. Algorithm 1 summarises the procedure.

Algorithm 1: Behavioural fingerprinting of tool-call trajectories

```

Input : corpus  $\{(\tau_i, y_i)\}$ ; representation  $R$  in  $\{\text{Struct}, \text{Verb}, \text{Behaviour}, \text{Content}, \text{Full}\}$ 
Output: fitted classifier  $f_R$  and cross-validated metric suite  $M$ 
1 for each trajectory  $\tau_i$ :
2   parse  $\tau_i \rightarrow [(o_t, s_t, p_t)]_{t=1..T}$  # typed actions, Eq.(1)
3    $\phi_{\text{struct}} \leftarrow$  16 dynamics features # length, entropy, retries, params Eq.(3-4)
4    $\phi_{\text{verb}} \leftarrow \text{tfidf}(1\text{-}3\text{gram over generic verbs})$  # workflow grammar
5    $\phi_{\text{content}} \leftarrow \text{tfidf}(1\text{-}2\text{gram over tool names})$  # Eq.(5)
6    $\phi_i \leftarrow \text{select/concat blocks per } R$  # Eq.(2)
7   standardise dense features; horizontally fuse sparse+dense blocks
8   for each model  $m$  in  $\{\text{LogReg}, \text{RF}, \text{ExtraTrees}, \text{HistGBM}\}$ :
9      $M[m] \leftarrow \text{StratifiedKFold}(5)\text{-CV}(m, \{\phi_i\}, \{y_i\})$  # Eq.(6-8)
10  return arg max over class posteriors, metric suite  $M$ 

```

4. Experimental Setup

4.1 Dataset

We evaluate on TRAJECT-Bench [8], a public benchmark of executable LLM-agent tool-use trajectories. We use the sequential configurations, retaining 1670 trajectories with non-empty tool lists across nine task domains. Each record provides the ordered list of tool calls with, for every call, the tool name, execution status, and the required and optional parameters supplied—exactly the typed-action structure of Eq. (1). One configuration (sequential e-mail) is malformed in the upstream repository and is excluded. Trajectories contain between 3 and 10 tool calls (median 6, mean 6.4). The prediction target is the task domain, giving a nine-class problem with a majority/random baseline of 0.111. Table 2 reports the class composition and Figure 2 visualises the per-domain counts and action-length distributions.

Table 2. Composition of the nine-domain trajectory corpus.

Domain	# Trajectories	Share
E-commerce	200	12.0%
Education	200	12.0%

Finance	135	8.1%
Gaming	185	11.1%
Mapping	200	12.0%
Music	200	12.0%
News/Media	180	10.8%
Travel	170	10.2%
Weather	200	12.0%
Total	1670	100.0%

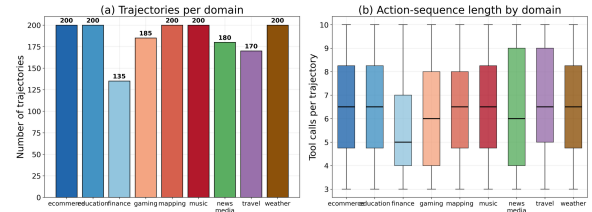


Figure 2. (a) Number of trajectories per domain. (b) Distribution of action-sequence lengths by domain; medians cluster near six calls, with domain-dependent spread.

4.2 Protocol

All metrics are computed under stratified five-fold cross-validation with a fixed random seed; out-of-fold predictions are aggregated before scoring so that every trajectory is evaluated exactly once out-of-sample. Confusion matrices and per-class metrics use these out-of-fold predictions. For efficiency profiling and the learning curve we additionally use a single stratified 75/25 split. TF-IDF vocabularies and the feature standardiser are fit within the training portion of each fold to prevent leakage. Statistical significance of differences between representations is assessed with paired t-tests over the five per-fold accuracies of a common classifier (extremely randomised trees). The complete pipeline—data download, feature extraction, training, and evaluation—is implemented as a single reproducible script using only standard scientific Python libraries [40].

5. Results and Performance Evaluation

5.1 The information gradient: how much fingerprint lives in each signal

Figure 3 and Table 3 report the central result. Moving from the weakest to the strongest representation, the best macro-F1 rises monotonically: Verb-ngram 0.857, Structural 0.968, Behaviour 0.995, and Content and Full both 1.000, against a random baseline of 0.111. The most important observation is that the Structural representation—sixteen features containing no tool identities, argument values, or textual tokens, already classifies the operational context with macro-F1 0.968, accuracy 0.967 ± 0.009 , Cohen’s κ 0.963, MCC 0.963, and one-vs-rest ROC-

AUC 0.999. Pure behavioural dynamics therefore constitute a strong fingerprint of what the agent is doing.

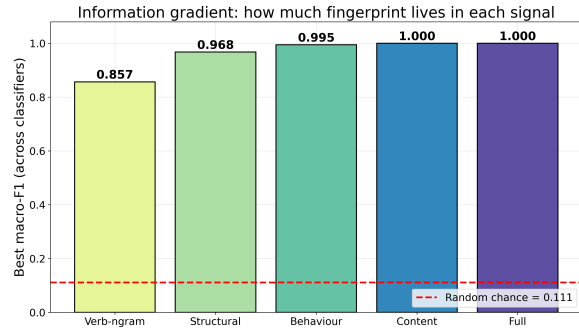


Figure 3. Information gradient: best macro-F1 (across classifiers) for each representation against the random-chance baseline. Purely structural dynamics reach 0.968 with no tool identities.

The increments are individually significant. Adding generic verb grammar to structural dynamics improves accuracy from 0.960 to 0.995 (paired t-test, $p = 0.0011$); behaviour exceeds the verb-only view by a wide margin ($p = 1e-05$); and the final gain from behaviour to content is small but significant ($p = 0.0086$). Both content-agnostic views vastly exceed chance (structural vs chance, $p = 3e-09$; verb vs chance, $p = 8e-09$). The interpretation is twofold. The near-perfect Content result is expected and, on its own, scientifically uninteresting: tool names are bound to domains (a Spotify search implies music, a hotel-pricing call implies travel), so classifying by tool vocabulary is close to a look-up. The meaningful findings are that (i) dynamics alone suffice for high accuracy and (ii) the tiny content-to-behaviour gap shows that most of the discriminative information is already present in behaviour, independent of identity.

Table 3. Best classifier per representation under 5-fold CV (mean±std for accuracy). Baseline = 0.111.

Representation	Best model	Accuracy	Macro-F1	Balance	Confusion	Mean C	ROC-AUC
Verb-ngram	ExtraTrees	0.866±0.009	0.857	0.856	0.849	0.850	0.993
Structural	HistGBM	0.967±0.009	0.968	0.968	0.963	0.963	0.999
Behaviour	ExtraTrees	0.995±0.002	0.995	0.994	0.994	0.994	0.999
Content	Log	1.000	1.0	1.	1.0	1.	1.

t	Reg	±0.000	00	00	00	00	00
Full	Log Reg	1.000±0.000	1.000	1.000	1.000	1.000	1.000

5.2 Classifier comparison and the role of non-linearity

Figure 4 shows macro-F1 for every representation-classifier pair. Two patterns stand out. On the Structural representation, logistic regression attains only 0.644 macro-F1 whereas the tree ensembles reach 0.968–0.962. This large gap confirms the design hypothesis of Section 3.3: behavioural dynamics separate contexts through non-linear interactions among features (e.g., retries combined with low success ratio and high parameter load), which a linear decision boundary cannot exploit. On content-bearing representations the distinction vanishes because tool-name n-grams are already linearly separable. Table 4 details the four classifiers on the Structural representation, the setting of greatest scientific interest.

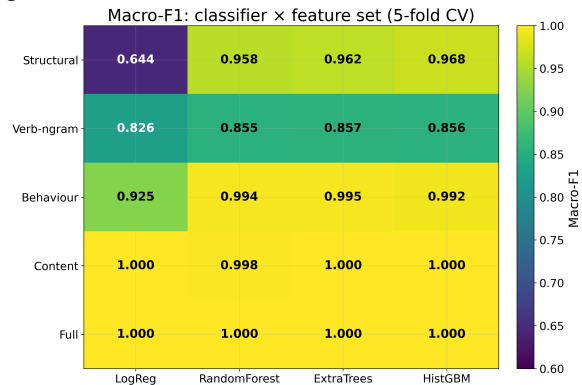


Figure 4. Macro-F1 for each classifier (columns) and representation (rows) under 5-fold CV. Tree ensembles dominate on the non-linear structural signal.

Table 4. Classifier comparison on the Structural (dynamics-only) representation.

Classifier	Accuracy	Macro-F1	Balance	Confusion	Mean C	ROC-AUC
LogReg	0.657±0.010	0.644	0.648	0.614	0.614	0.927
Random Forest	0.957±0.006	0.958	0.958	0.951	0.955	0.998
ExtraTrees	0.960±0.008	0.962	0.962	0.955	0.955	0.997
HistGBM	0.967±0.009	0.968	0.968	0.963	0.963	0.999

5.3 Per-class behaviour and error structure

Figure 5 presents the out-of-fold confusion matrices for the two content-agnostic detectors. The Structural-only matrix is strongly diagonal: every domain is recovered with per-class F1 between 0.947 and 0.990 (Table 5). Residual confusions are semantically plausible mapping is occasionally mistaken for music or news/media, domains whose look-up-style, low-parameter cadence is genuinely similar rather than random, which further supports the claim that the model captures behavioural structure. Adding verb grammar (the Behaviour matrix) removes almost all residual error.

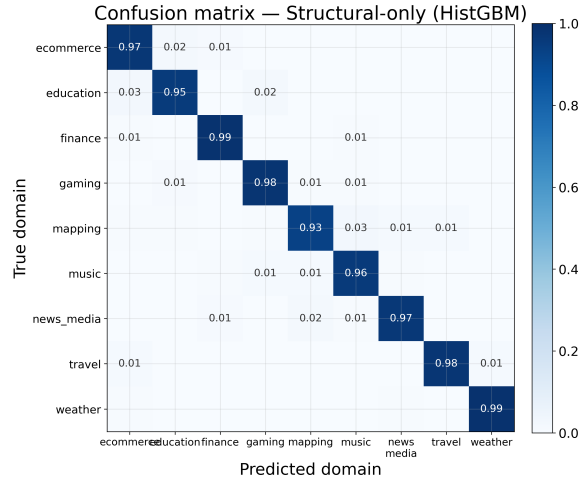


Figure 5a. Confusion matrix (row-normalised) for the Structural-only detector (HistGBM), aggregated over 5-fold out-of-fold predictions.

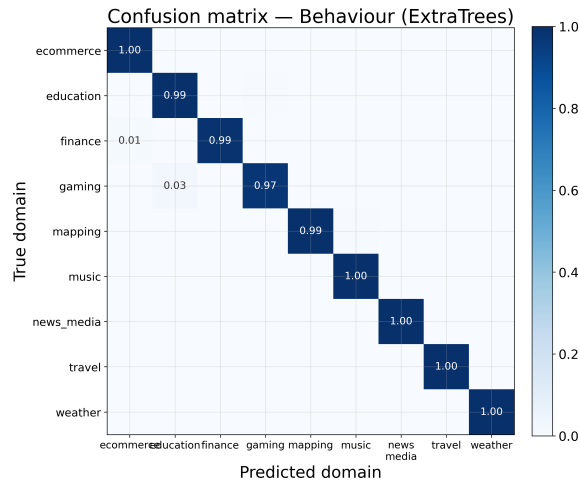


Figure 5b. Confusion matrix for the Behaviour detector (ExtraTrees). Adding generic action-verb grammar to dynamics resolves nearly all residual confusions.

Table 5. Per-domain performance of the Structural-only detector (out-of-fold, HistGBM).

Domain	Precisio	Recal	F1	Support
--------	----------	-------	----	---------

	n	l		t
E-commerce	0.951	0.970	0.960	200
Education	0.965	0.955	0.960	200
Finance	0.964	0.985	0.974	135
Gaming	0.958	0.978	0.968	185
Mapping	0.964	0.930	0.947	200
Music	0.960	0.960	0.960	200
News/Media	0.972	0.967	0.969	180
Travel	0.982	0.976	0.979	170
Weather	0.990	0.990	0.990	200

5.4 Which behaviours carry the signature?

Figure 6 reports impurity-based and permutation importances for the sixteen structural features. Both agree on the dominant signals. The most informative features are verb_entropy (0.132), avg_req (0.112), max_opt (0.105), std_req (0.101), param_load (0.093). That verb-entropy, parameter-load statistics (average and maximum required/optional parameters, and their variability), and total parameter load dominate is behaviourally interpretable: domains differ chiefly in how exploratory their action distribution is and in how heavily parameterised their calls are weather look-ups are terse and low-parameter, whereas financial or travel workflows chain heavier, more variable calls. Permutation importance, which measures the accuracy drop when a feature is shuffled on held-out data (Eq. (10)), confirms the ranking and shows the effect is not an artefact of tree-split counting.

$$\text{Imp}(x_j) = \text{Acc}(\phi) - \mathbb{E}_{\pi}[\text{Acc}(\phi \mid x_j \rightarrow a)] \quad (10)$$

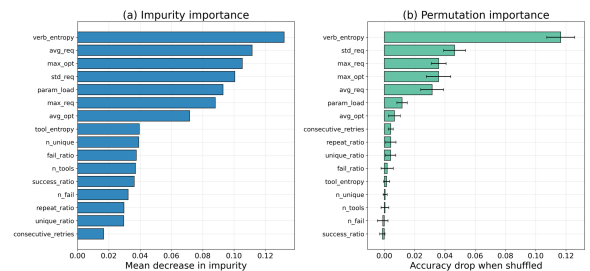


Figure 6. Structural feature importance: (a) mean decrease in impurity; (b) permutation importance (accuracy drop on held-out data, with standard deviation). Parameter-complexity and entropy features dominate.

5.5 Data efficiency and generalisation

Figure 7 shows the learning curve of the Structural-only detector. Cross-validated accuracy rises from 0.120 with 133 training trajectories to 0.967 with the full training set, with the training validation gap narrowing as data grow evidence of good generalization and only mild over-fitting for a sixteen-feature model. The curve is still gently rising at the largest size, indicating that additional trajectories would yield further, if modest, gains.

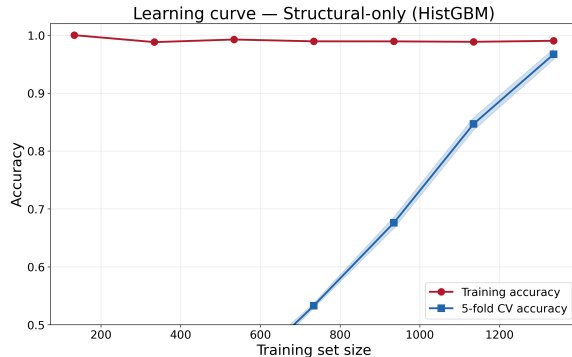


Figure 7. Learning curve for the Structural-only detector (HistGBM); shaded band is ± 1 standard deviation of the cross-validated accuracy.

5.6 Early detection from partial trajectories

A practical detector must act before a trajectory completes. We therefore truncate each trajectory to its first k actions, recompute the structural features on the prefix (Eq. (9)), and re-evaluate. Figure 8 reports the result. From a single observed action, the detector already reaches 0.406 accuracy roughly four times chance and it crosses 90% by $k = 4$ actions, plateauing near 0.968. Because the median trajectory contains six calls, four to five actions represent well under a full trajectory. This establishes that behavioural fingerprints are available early, which is the regime relevant to online monitoring and to intervening before a potentially harmful action sequence completes.

$$\text{Acc}(k) = \mathbb{E}_{\tau}[1[\hat{y}(\phi(\tau_{1:k})) = y(\tau)]], \quad k = 1, 2, \quad (9)$$

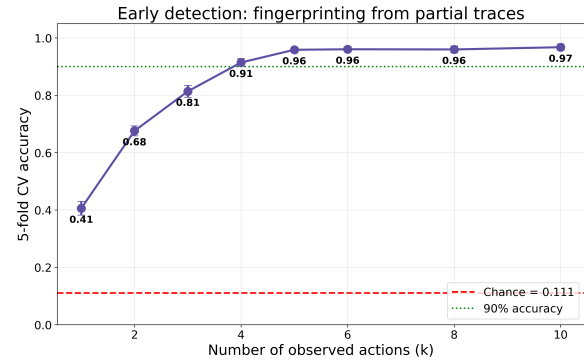


Figure 8. Early detection: 5-fold CV accuracy of the Structural-only detector as a function of the number of observed actions k . Four to five actions suffice for $>90\%$ accuracy.

5.7 Computational efficiency

Table 6 profiles training cost and inference latency. The Structural-only detector is inexpensive: it trains in 2.16 s and classifies a trajectory in 0.16 ms on commodity hardware using only sixteen features, while retaining held-out accuracy of 0.952. Figure 9 plots held-out accuracy against training time for all representation-model pairs and shows that the dynamics-only detectors occupy the favourable low-cost, high-accuracy region. For deployment as an always-on observability filter, the structural detector is thus attractive: sub-millisecond latency and a tiny feature footprint, at a small accuracy cost relative to the content model that would in any case be trivially circumventable by tool renaming.

Table 7. Training cost and inference latency (single 75/25 split).

Represent ation	Model	# Fe at.	Fit (s)	Laten cy (ms/tr aj)	Hel d-out Acc
Structural	LogReg	16	0.0 6	0.002	0.6 48
Structural	RandomF orest	16	0.5 4	0.072	0.9 43
Structural	ExtraTree s	16	0.4 2	0.072	0.9 45
Structural	HistGBM	16	2.1 6	0.159	0.9 52
Full	LogReg	566	0.1 5	0.002	1.0 00
Full	RandomF orest	566	0.6 2	0.074	1.0 00
Full	ExtraTree s	566	0.5 8	0.083	1.0 00
Full	HistGBM	566	17. 34	0.142	1.0 00

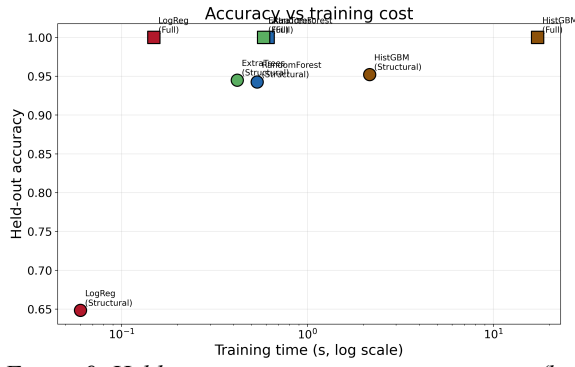


Figure 9. Held-out accuracy versus training time (log scale) for every representation-model pair; marker shape encodes representation.

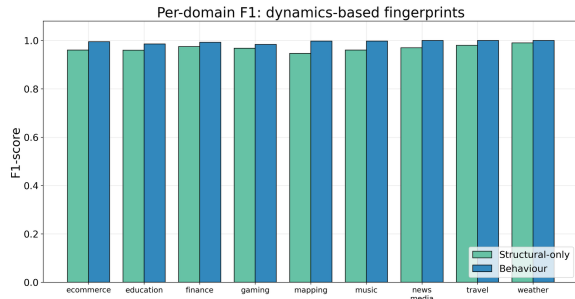


Figure 10. Per-domain F1 for the two content-agnostic detectors; dynamics alone already exceed 0.94 F1 in every domain.

6. Comparison with Recent Works

No prior method performs behavioural fingerprinting of autonomous agents from tool-call dynamics, so a like-for-like numerical comparison is not meaningful: related methods solve different tasks on different inputs, and reporting their headline accuracies side by side would be misleading. We therefore compare capabilities qualitatively in Table 8 and report quantitative numbers only for our own method, where they are directly measured. The comparison spans the three most closely related paradigms: output-based LLM fingerprinting, system-call-based host intrusion detection, and trajectory-based agent evaluation.

Table 7. Capability comparison with the most closely related paradigms. Dashes denote properties that are not applicable or not comparable across tasks; numerical performance is reported only where we measured it directly.

Meth od / para digm	Inp ut mod ality	Nee ds cont ent / iden tity?	Nee ds mod el inter nals or quer y	Tas k	Ear ly dete ction	Our meas ured perfo rman ce

			acce ss?			
LLM map [15]	mod el resp onse s to craft ed prob es	Yes (gen erate d text)	Yes (acti ve queri es)	Ident ify mod el fami ly	—	—
Instru ction al FP [16]	emb edde d trigg er resp onse s	Yes	Yes (own er- emb edde d)	IP / own ershi p	—	—
HuRe f [17]	mod el weig hts	—	Yes (whit e- box)	Own ershi p finge rprin t	—	—
Sysca ll HIDS [25, 26]	OS syst em- call sequ ence	Yes (call iden tity)	No	Proc ess anoma ly	Part ial	—
Traje ctory eval [8, 10]	tool- call traje ctor y	Yes	Som etim es	Scor e task succ ess	No	—
Ours — Struct ural	tool- call dyna mics only	No	No (pass ive)	Ident ify agen t cont ext	Yes (k= 4–5)	0.968 macro -F1
Ours — Beha viour	dyna mics + gene ric verb s	No	No (pass ive)	Ident ify agen t cont ext	Yes	0.995 macro -F1

The comparison clarifies the contribution. Relative to output-based LLM fingerprinting [15, 16, 17, 18, 20], our method is passive and content-independent: it

needs neither query access, owner-embedded triggers, nor model weights, and it operates on a modality (tool-call dynamics) that those methods do not use. Relative to system-call HIDS [24, 25, 26], which is the closest methodological ancestor, we transfer the sequence-classification premise to a new domain and, crucially, go further by isolating identity-free structural dynamics rather than relying on call identity. Relative to trajectory-evaluation benchmarks [8, 10, 11, 12], we repurpose the trajectory from a correctness artefact into an identity signal and add an early-detection capability absent from that literature. The performance improvements we claim are therefore not “higher accuracy on the same task,” but the demonstration that a strictly weaker, cheaper, and more tamper-resistant input behavioural dynamics with no identifying content is sufficient for high-accuracy, early identification, a capability no prior paradigm provides.

7. Discussion and Limitations

Scope of the label. The most important caveat is that the recovered label is the task domain, not the backbone model. Our results demonstrate that behavioural dynamics fingerprint what operational context an agent is acting in; they do not, on this corpus, demonstrate attribution of which underlying LLM produced the trajectory, because no public corpus yet labels tool-call traces by backbone model. We are deliberate about this distinction to avoid over-claiming. The method and the factorised feature design transfer directly to model attribution once model-labelled traces exist, and we regard that as the natural next study rather than a completed result.

Triviality of content and the value of dynamics. The Content and Full representations reach perfect accuracy, but this is an artefact of tool names being bound to domains and is not evidence of a difficult problem solved. We have been careful to frame the scientific contribution around the content-agnostic results the structural and behaviour representations—and the early-detection analysis, precisely because those cannot be explained by identity look-up. A corollary is a security argument: a content-based detector is trivially defeated by renaming tools, whereas a dynamics-based detector keys on properties (retry structure, parameter load, entropy) that an agent must exhibit to accomplish its task and cannot as easily disguise.

Single-corpus evaluation and generalisation. All experiments use one benchmark [8]. Although it is executable, multi-domain, and grounded in production-style APIs, cross-corpus generalisation to other agent ecosystems, tool taxonomies, and real deployment logs is untested and is a threat to external validity. The domains are also curated and largely non-overlapping; real deployments will contain

ambiguous or mixed-context sessions that are inherently harder.

Adversarial robustness. Our threat model is observational, not adversarial. A motivated agent could attempt to evade a dynamics-based detector by padding, reordering, or throttling its actions. Because several discriminative features (parameter load, success profile) are tied to task requirements, such evasion is likely to incur a utility cost, but quantifying the robustness–utility trade-off requires a dedicated adversarial study. Finally, the structural features assume the tool layer logs execution status and parameter counts; environments that expose only opaque tool names would fall back to the weaker verb-only view.

8. Conclusion and Future Work

We introduced behavioural fingerprinting of autonomous LLM agents from the dynamics of their tool-call trajectories and showed, on a public benchmark of 1670 executable trajectories across nine domains, that the shape of an agent’s actions is a strong and cheap identity signal. A representation of sixteen content-agnostic dynamics features carrying no tool identities whatsoever identifies the operational context with macro-F1 0.968 against a 0.111 baseline, adding generic action grammar raises this to 0.995, and every step of the information gradient is statistically significant. Four to five observed actions suffice for over 90% accuracy, and the detector runs in well under a millisecond per trajectory. Together these establish that behaviour, independent of content, fingerprints an agent’s context, a result with direct application to agent observability, anomaly detection, and provenance in security-sensitive deployments [32, 33, 34].

Future work proceeds along four axes. First, backbone-model attribution: applying the identical pipeline to traces labelled by the generating model to test whether dynamics also fingerprint the model, not only the task. Second, adversarial robustness: characterising evasion strategies and hardening the detector, drawing on the agent-security literature [35, 36, 38]. Third, richer temporal models: replacing aggregate features with sequence models that consume the ordered action stream and incorporate inter-call timing, extending the sequence-classification tradition of host-based detection [28, 31]. Fourth, cross-corpus and streaming evaluation: validating on independent agent logs and in an online setting with concept drift. We release the complete, single-file benchmark to support reproduction and these extensions.

References

- [1] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, W. X. Zhao, Z. Wei, and J.-R. Wen, "A survey on large language model based autonomous agents," *Frontiers of Computer Science*, vol. 18, no. 6, art. 186345, 2024.
- [2] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, et al., "The rise and potential of large language model based agents: A survey," *arXiv preprint arXiv:2309.07864*, 2023.
- [3] C. Qu, S. Dai, X. Wei, H. Cai, S. Wang, D. Yin, J. Xu, and J.-R. Wen, "Tool learning with large language models: A survey," *Frontiers of Computer Science*, vol. 19, no. 8, 2025.
- [4] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, et al., "AutoGen: Enabling next-gen LLM applications via multi-agent conversation," *arXiv preprint arXiv:2308.08155*, 2023.
- [5] T. Schick, J. Dwivedi-Yu, R. Dessi, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023.
- [6] S. G. Patil, T. Zhang, X. Wang, and J. E. Gonzalez, "Gorilla: Large language model connected with massive APIs," *arXiv preprint arXiv:2305.15334*, 2023.
- [7] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, et al., "ToolLLM: Facilitating large language models to master 16000+ real-world APIs," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2024.
- [8] P. He, Z. Dai, B. He, H. Liu, X. Tang, H. Lu, J. Li, J. Ding, S. Mukherjee, S. Wang, Y. Xing, J. Tang, and B. Dumoulin, "TRAJECT-Bench: A trajectory-aware benchmark for evaluating agentic tool use," *arXiv preprint arXiv:2510.04550*, 2025.
- [9] S. Yao, N. Shinn, P. Razavi, and K. Narasimhan, "tau-bench: A benchmark for tool-agent-user interaction in real-world domains," *arXiv preprint*, 2024.
- [10] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, et al., "AgentBench: Evaluating LLMs as agents," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2024.
- [11] J. Lu, T. Holleis, Y. Zhang, B. Aumayer, F. Nan, F. Bai, et al., "ToolSandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities," *arXiv preprint arXiv:2408.04682*, 2024.
- [12] Z. Wang, Q. Chang, H. Patel, S. Biju, C.-E. Wu, Q. Liu, et al., "MCP-Bench: Benchmarking tool-using LLM agents with complex real-world tasks via MCP servers," *arXiv preprint arXiv:2508.20453*, 2025.
- [13] Y. Song, D. Yin, X. Yue, J. Huang, S. Li, and B. Y. Lin, "Trial and error: Exploration-based trajectory optimization of LLM agents," in *Proc. 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 7584-7600, 2024.
- [14] A. Nasery, et al., "Scalable fingerprinting of large language models," *arXiv preprint arXiv:2502.07760*, 2025.
- [15] D. Pasquini, E. M. Kornaropoulos, and G. Ateniese, "LLMmap: Fingerprinting for large language models," in *Proc. 34th USENIX Security Symposium*, pp. 299-318, 2025.
- [16] J. Xu, F. Wang, M. D. Ma, P. W. Koh, C. Xiao, and M. Chen, "Instructional fingerprinting of large language models," in *Proc. Conf. of the North American Chapter of the ACL (NAACL-HLT)*, pp. 3277-3306, 2024.
- [17] Y. Hu, Z. Lin, L. Wang, X. Wang, Y. Xu, Y. Yu, B. Zeng, and C. Zhou, "HuRef: Human-readable fingerprint for large language models," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 37, 2024.
- [18] M. Gubri, D. Ulmer, H. Lee, S. Yun, and S. J. Oh, "TRAP: Targeted random adversarial prompt honeypot for black-box identification," in *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11496-11517, 2024.
- [19] H. Jin, C. Zhang, S. Shi, W. Lou, and Y. T. Hou, "ProFLingo: A fingerprinting-based intellectual property protection scheme for large language models," in *Proc. IEEE Conf. on Communications and Network Security (CNS)*, pp. 1-9, 2024.
- [20] D. Bhardwaj and N. Mishra, "Invisible traces: Using hybrid fingerprinting to identify underlying LLMs in GenAI apps," *arXiv preprint arXiv:2501.18712*, 2025.
- [21] Y. Bitton, E. Bitton, and S. Nisan, "Detecting stylistic fingerprints of large language models," *arXiv preprint arXiv:2503.01659*, 2025.
- [22] Z. Xu, M. Han, and W. Xing, "EverTracer: Hunting stolen large language models via stealthy and robust probabilistic fingerprint," in *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7019-7042, 2025.
- [23] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, "A watermark for large language models," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2023.
- [24] Z. T. Sworna, Z. Mousavi, and M. A. Babar, "NLP methods in host-based intrusion detection

- systems: A systematic review and future directions," *Journal of Network and Computer Applications*, vol. 220, art. 103761, 2023.
- [25] B. Subba and P. Gupta, "A TfIdfVectorizer and singular value decomposition based host intrusion detection system framework for detecting anomalous system processes," *Computers & Security*, vol. 100, art. 102084, 2021.
- [26] Y. Kumar and B. Subba, "Stacking ensemble-based HIDS framework for detecting anomalous system processes in Windows based operating systems using multiple word embedding," *Computers & Security*, vol. 125, art. 102961, 2023.
- [27] P. K. Mvula, P. Branco, G.-V. Jourdan, and H. L. Viktor, "Evaluating word embedding feature extraction techniques for host-based intrusion detection systems," *Discover Data*, vol. 1, no. 1, art. 2, 2023.
- [28] N. Joraviya, B. N. Gohil, and U. P. Rao, "DL-HIDS: deep learning-based host intrusion detection system using system calls-to-image for containerized cloud environment," *The Journal of Supercomputing*, vol. 80, no. 9, pp. 12218-12246, 2024.
- [29] S. Wunderlich, M. Ring, D. Landes, and A. Hotho, "The impact of different system call representations on intrusion detection," *Logic Journal of the IGPL*, vol. 30, no. 2, pp. 239-251, 2022.
- [30] X. Liao, C. Wang, and W. Chen, "Anomaly detection of system call sequence based on dynamic features and relaxed-SVM," *Security and Communication Networks*, vol. 2022, art. 6401316, 2022.
- [31] M. Nakip and E. Gelenbe, "Online self-supervised deep learning for intrusion detection systems," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 5668-5683, 2024.
- [32] E. Debenedetti, J. Zhang, M. Balunovic, L. Beurer-Kellner, M. Fischer, and F. Tramer, "AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents," in *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, vol. 37, 2024.
- [33] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not what you have signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection," in *Proc. ACM Workshop on Artificial Intelligence and Security (AISec)*, 2023.
- [34] Y. Gan, Y. Yang, Z. Ma, P. He, R. Zeng, Y. Wang, et al., "Navigating the risks: A survey of security, privacy, and ethics threats in LLM-based agents," *arXiv preprint arXiv:2411.09523*, 2024.
- [35] M. A. Ferrag, N. Tihanyi, D. Hamouda, L. A. Maglaras, and M. Debbah, "From prompt injections to protocol exploits: Threats in LLM-powered AI agents workflows," *arXiv preprint arXiv:2506.23260*, 2025.
- [36] S. Vimal, V. Jeyabalaraja, P. Subbulakshmi, A. Suresh, M. Kaliappan, Deep learning-based decision-making with WoT for smart city development, *Smart innovation of web of things*, CRC Press, Boca Raton, 51-62, 29, 2020.
- [37] B. Paramasivan, M. Kaliappan, Secure and fair cluster head selection protocol for enhancing security in mobile Ad hoc networks, *The Scientific World Journal*, 29, 2014.
- [38] G S, RB J, HR Y, R S, V S, D Pelusi, Kaliappan M, Geo Spatial Based Real Time Monitoring on Eutrophic Evaluation of Porunai River Basin for Pollution Risk Assessment, *European Journal of Remote Sensing*, 23, 2022.
- [39] S. Vimal, M. Kaliappan, A. Suresh, P. Subbulakshmi, S. Kumar, D. Kumar, Development of Cloud Integrated Internet of Things Based Intruder Detection System, *Journal of Computational and Theoretical Nanoscience* 15 (11-12), 3565-3570, 22, 2018.
- [40] C. Karpagavalli, M. Kaliappan, Edge Implicit Weighting with Graph Transformers for Robust Intrusion Detection in Internet of Things Network, *Computers & Security*, 21, 2025.
- [41] M. Kaliappan, E. Mariappan, B. Revathi, Unified Vision Transformer for Multimodal Medical Image Classification, *Biomedical Signal Processing and Control* 110431, 20, 2026.
- [42] Ranjit, K. C., Shrestha, B., & Ghimire, U. (2026). Optimizing CNN Architectures for Steering Angle Prediction for Self-Driving Vehicles in Unstructured Roads: A Comparative Study of Activation Functions and Model Complexity. *ICCK Transactions on Machine Intelligence*, 2(2), 88–99. <https://doi.org/10.62762/TMI.2025.759110>
- [43] Kumar, A., & Lehal, G. S. (2026). Detection of Newspaper Layouts Using YOLO12. *ICCK Transactions on Machine Intelligence*, 2(2), 77–87. <https://doi.org/10.62762/TMI.2025.846033>
- [44] Wang, X., & Zhao, R. (2026). FedTLRec: Federated Recommendation with Transformer-based Parameter Aggregation and LoRA Compression. *ICCK Transactions on Machine Intelligence*, 2(1), 65–76. <https://doi.org/10.62762/TMI.2025.882476>

- [45] Arya, S., Anju, & Onah, J. N. (2026). A Data-Driven Framework for Methane Emission Prediction Using Machine Learning Methods. *ICCK Transactions on Machine Intelligence*, 2(1), 53–64. <https://doi.org/10.62762/TMI.2025.782852>
- [46] Bai, Y., Hu, J., Tursun, E. & Yimit, H. (2026). Data-Driven Operational Assessment Method and Digital Twin System for Unmanned Surface Vehicles. *ICCK Transactions on Machine Intelligence*, 2(1), 38–52. <https://doi.org/10.62762/TMI.2025.444910>
- [47] Agarwal, S. K., & Gupta, Y. K. (2026). A Novel Approach of Progressive Transfer Learning for MRI Brain Tumor Classification Using VGG16 and MobileNet Architectures. *ICCK Transactions on Machine Intelligence*, 2(1), 28–37. <https://doi.org/10.62762/TMI.2025.367009>
- [48] Kumar, V., Saxena, A., & Chawla, L. (2026). Adaptive Learning Density Estimators for Tsallis Entropy and Kapur Entropy with Applications in System Training. *ICCK Transactions on Machine Intelligence*, 2(1), 12–27. <https://doi.org/10.62762/TMI.2025.317970>
- [49] Agarwal, S., Kandpal, M., Gupta, V. K., Aggarwal, S., & Jain, P. (2025). Secure and Decentralized Heart Sound Analysis using Federated Learning and Blockchain Technology. *ICCK Transactions on Machine Intelligence*, 2(1), 1–11. <https://doi.org/10.62762/TMI.2025.567350>
- [50] Pereira, M. S., de Azevedo, T. M., Ribeiro, C. T., Freire, A. I., Francisco, M. B., Pereira, J. L. J., & de Paiva, A. P. (2025). Discriminating Planted Capsicum Spp. Varieties via Machine Learning and Multivariate Data Reduction. *ICCK Transactions on Machine Intelligence*, 1(3), 166–185. <https://doi.org/10.62762/TMI.2025.385133>
- [51] Kumar, A. (2025). Clinical Text Analytics: Techniques, Deep Learning Models, and the Future of Medical Text Analytics. *ICCK Transactions on Machine Intelligence*, 1(3), 148–165. <https://doi.org/10.62762/TMI.2025.451731>
- [52] Malik, I. A., & Habib, G. (2025). AI Enabled Resource-Constrained Computing Architectures for IoT Devices. *ICCK Transactions on Machine Intelligence*, 1(3), 138–147. <https://doi.org/10.62762/TMI.2025.225921>
- [53] Bmotra, M., Kumar, R., & Dhiman, G. (2025). Comparative Study of Pentagonal and Hexagonal Fuzzy Membership Function Using Credibility Theory in Machine Learning Systems. *ICCK Transactions on Machine Intelligence*, 1(3), 127–137. <https://doi.org/10.62762/TMI.2025.922612>
- [54] Aggarwal, V., Aggarwal, S., Bhardwaj, A., & Gupta, V. K. (2025). Quantum Computing Essentials: Bridging Theory and Practice for New Learners. *ICCK Transactions on Machine Intelligence*, 1(3), 117–126. <https://doi.org/10.62762/TMI.2025.173543>
- [55] Sarvakar, K., & Rana, K. (2025). A Hybrid Framework Combining CNN, LSTM, and Transfer Learning for Emotion Recognition. *ICCK Transactions on Machine Intelligence*, 1(2), 103–116. <https://doi.org/10.62762/TMI.2025.572412>
- [56] Singh, S., Aggarwal, S., Singh, A., & Sharma, A. (2025). Integrating Artificial Intelligence and Machine Learning in Autism Detection via Gut Microbiome Analysis. *ICCK Transactions on Machine Intelligence*, 1(2), 90–102. <https://doi.org/10.62762/TMI.2025.682666>
- [57] Sharma, S., Devgan, K., Jangra, D., Bhardwaj, A., & Aggarwal, S. (2025). Neuro-Inspired Alert System for Air Quality Prediction Using Ensemble Preprocessing and SNN Classification. *ICCK Transactions on Machine Intelligence*, 1(2), 69–79. <https://doi.org/10.62762/TMI.2025.403059>
- [58] Khan, A., & Singh, J. (2025). A Novel Image Captioning Technique Using Deep Learning Methodology. *ICCK Transactions on Machine Intelligence*, 1(2), 52–68. <https://doi.org/10.62762/TMI.2025.886122>
- [59] Khan, M., Kumar, R., & Dhiman, G. (2025). A Hybrid Machine Learning Fuzzy Non-linear Regression Approach for Neutrosophic Fuzzy Set. *ICCK Transactions on Machine Intelligence*, 1(1), 42–51. <https://doi.org/10.62762/TMI.2025.561363>
- [60] Garg, M. (2025). Enhanced Reinforcement Learning-Based Resource Scheduling for Secure Blockchain Networks in IIoT. *ICCK Transactions on Machine Intelligence*, 1(1), 29–41. <https://doi.org/10.62762/TMI.2024.529242>
- [61] Gupta, S., Adhikari, U., Roy, D., & Hazra, S. (2025). IoT-Integrated Reinforcement Learning-Based Mine Detection System for Military and Humanitarian Applications. *ICCK Transactions on Machine Intelligence*, 1(1), 17–28. <https://doi.org/10.62762/TMI.2025.235880>
- [62] Myakala, P. K., Kamatala, S., & Bura, C. (2025). Privacy-Preserving Federated Learning for IoT Botnet Detection: A Federated Averaging Approach. *ICCK Transactions on Machine Intelligence*, 1(1), 6–16. <https://doi.org/10.62762/TMI.2025.796490>

- [63] Tiwari, S., Kumar, D., Mahajan, A., & Sachar, S. (2025). Emotion Detection from Speech Using CNN-BiLSTM with Feature Rich Audio Inputs. *ICCK Transactions on Machine Intelligence*, 1(2), 80–89. <https://doi.org/10.62762/TMI.2025.306750>