Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Auditable Tool Selection for Fraud-Related Website Investigation

qiang fu
Criminal Investigation Police University of China
JunKun Li
198743468100163.com

Criminal Investigation Police University of China

Research Article

Keywords: fraud-related websites, tool selection, evidence provenance, security investigation, symbolic world models

Posted Date: November 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11690710/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

International Journal of Information Security manuscript No.
(will be inserted by the editor)

Auditable Tool Selection for Fraud-Related Website Investigation

Qiang Fu · Junkun Li

Abstract When a fraud-related website becomes unavailable, investigators must decide whether to recheck its domain, trace a redirect, or compare an associated application. A failed request does not establish that the underlying activity has ceased. We propose an auditable procedure for selecting verification actions under a fixed tool-call budget. Each call executes one action. A symbolic model specifies competing explanations and their expected outputs. The selector maximises the number of explanation pairs an action could distinguish per unit assigned cost. Responses eliminate incompatible explanations before scores are recalculated; unresolved propositions remain Unknown. Observations retain their sources and acquisition conditions. We evaluate evidence acquisition separately from action selection. Across 87 retrieval queries, Recall@5 is 0.7671 for BM25 and 0.8868 for rank fusion with cross-encoder reranking. Two related Android application builds have matching content for 99.90% of their entries, illustrating resource-reuse evidence obtainable through application comparison. In retrospective replays, completion requires agreement with all scored reference labels and supporting evidence files that pass provenance checks. Under a two-call budget, the selector completes all 12 tasks in the 12-task set, against 4 and 3 under fixed and heuristic policies. With the same budget in the 27-task set, it completes 25 tasks in 25 calls and returns Unknown on 2; the baselines complete all 27 using 42 and 40 calls. The procedure records why each action was selected and how its return affects the remaining explanations. The small number of source cases limits generalisation to new investigations.

Qiang Fu (corresponding author) and Junkun Li
Criminal Investigation Police University of China, No. 83 Tawan Street, Huanggu District, Shenyang 110854, Liaoning, China
E-mail: 19874346810@163.com

Keywords fraud-related websites · tool selection · evidence provenance · security investigation · symbolic world models

1 Introduction

Tracing a fraud-related website is a chain of small decisions: which address to open, which redirect to follow, which sample to compare. Each element moves on its own schedule. An entry domain stops resolving while a redirect endpoint stays up; the same content reappears at a new address. The investigator's real question is therefore narrow: what could this tool return, and would that return rule out an explanation still on the table?

Open-source intelligence research asks how public information can inform a cybersecurity investigation [28], and anti-phishing reviews map detection and prevention across attack channels [26]. Surveys of large language models (LLMs) in cybersecurity describe their use in detection and defence [16], while detectors trained on one region's phishing corpus can lose accuracy on another's [31]. For website verification the lesson is concrete: a shared address may mean shared hosting, and identical resources may mean template reuse. An action is worth taking when it supplies the evidence the association is still missing.

The system we study already supports collection, extraction, graph construction, sample parsing and retrieval-based question answering. Its traces record which tools ran, but not which alternatives or expected outcomes were considered before each call. Security surveys treat an agent's tool calls as an audit surface [27]; our formulation of action selection targets exactly that missing record.

Here, a tool call is one verification action submitted through the available tool interfaces. The replay program selects actions automatically and reveals their archived returns under a fixed call budget. The budget is an experimental constraint used to compare action orders; it is not a claim about a universal operational quota. The study evaluates action selection, rather than an end-to-end autonomous investigation system.

World models give an agent a basis for prediction, planning and feedback from observation [6,10,11]. In website verification, most tools change what the investigator knows about an object, not the object itself. Our formulation separates external change — entry migration, infrastructure reuse and content reuse — from the observations returned by a DNS query, redirect trace or Android application package (APK) extraction. Candidate returns are then compared against those explanations.

We formulate website verification as choosing observations from a finite action set. The world model links a provenance- and condition-aware evidence state to competing scenarios and expected tool returns, while proposition dependencies define the scope of revision. Existing retrieval, graph-query and sample-analysis components supply observations; the replay implementation evaluates budget-aware action selection under shared returns.

The central contribution is an auditable chain from evidence state to action choice and proposition-scoped revision. Section 3 formalises observations, relations, propositions and unresolved gaps together with their provenance, time and acquisition conditions (Table 2). The empirical study tests the observation pipeline through six retrieval configurations on 87 queries, content comparison across three Android builds and trace reconstruction (Sections 5.1–5.6), then compares explicit action-selection rules with ReAct and Reflexion under shared returns and bounded call budgets (Sections 4.5 and 5.7; Supplementary Appendix B).

Three questions organize the evaluation. RQ1 asks how retrieval and sample comparison provide useful, traceable observations. RQ2 asks which acquisition and retention conditions make tool outcomes and evidence-linked revisions reconstructable. RQ3 asks how action-selection rules allocate a constrained call budget under shared returns, including comparison with LLM selectors. Component scores answer RQ1, trace and case analyses answer RQ2, and the replay answers RQ3.

2 Related Work and Methodological Foundations

2.1 Website Associations and Evidence Levels

Website associations form at three layers: entry, infrastructure and content. A shared service links objects with nothing else in common, while an uncommon distribution identifier inside a complete redirect chain can narrow the candidate set sharply. Following provenance-sensitive open-source intelligence work [9], we keep infrastructure links, distribution links and resource reuse separate from the operator relationship, which always needs further verification. Two recent lines run close to this problem: explainable feature selection combined with phishing-website classifiers [2], and multi-agent LLM pipelines for phishing detection [29].

World-model research ties an internal state representation to predictions that inform decisions [6]. We focus on a specific decision: which verification action would resolve the association, and what a DNS recheck, redirect trace or sample comparison could add.

2.2 World Models and Tool-Action Planning

Ding et al. separate internal world representations from simulations of future states [6], and Dreamer learns latent dynamics for imagined control [10,11]. A recent review of edge intelligence examines how world models support prediction and planning alongside language-based agents [36], while ReAct, RAP and Reflexion treat reasoning, planning and feedback from different angles [12,23,30]. We take one rule from this work: weigh an action's expected outcome against the evidence already in hand. Surveys of tool learning describe how language models select and invoke external tools [21], and reliability harnesses test whether those calls execute as intended [4].

We use world model in the model-based sense of an internal predictive representation [6,10–12]. Here the representation is symbolic: candidate scenarios specify observable tool returns under acquisition conditions, and the selector measures how strongly those returns distinguish the scenarios. A versioned evidence state and dependency-scoped revision connect each selected observation to the propositions it can update.

2.3 Provenance, Retrieval and Action Traceability

PROV models entities, activities and agents [19], and fact-verification research separates supporting, refuting and insufficient evidence [24]. Our schema records call

parameters, the location of the original result, and the conditions under which we acquired it. Forensic provenance analysis reconstructs how an exhibit was produced and edited [34]; we keep the same kind of record for tool returns. Reposts and summaries point back to their original sources, so repeated publication never counts as independent support, and separate acquisitions keep their own provenance and conditions.

Retrieval-augmented generation grounds responses in retrieved documents [18]. We draw candidates from BM25 [22] and dense retrieval, fuse the rankings with reciprocal rank fusion (RRF) [5], and rerank query–document pairs with a cross-encoder (CE) implemented as bge-reranker-v2-m3 [3]. A recent GraphRAG survey separates graph-based indexing, graph-guided retrieval and graph-enhanced generation [20]; another line gates the flow of evidence along graph relations during retrieval [35], and an information-theoretic framework treats retrieval-augmented generation as evidence acquisition [32]. Section 3.1 describes our bounded two-hop branch. We compare configurations on paired queries and read the differences against evaluation guidance [8]; Table 1 relates these components to the planning layer.

3 A World-Model-Based Framework for Tool-Based Verification

3.1 Overall Architecture and Computational Workflow

Figure 1 connects four stages: evidence acquisition, world-model planning, tool execution and evidence revision. Panel (a) brings collection and grading (E1) together with retrieval and source review (E3) to populate B(k). Panel (b) turns the gaps U(k) into a candidate action set A(k): the change model T and scenarios Σ describe possible external states, G predicts observable tool fields under conditions C, and the planner compares their distinguishing value and selects q* using Equation (4). Panel (c) routes that action to an available interface and keeps its call and return record (E4–E5). Panel (d) checks the observation conditions (E2), compares the return with the prediction and revises the affected propositions. The migration analysis (E6) reconstructs this last stage from archived records.

The framework exposes why each verification action is selected. Before a call, the agent records the assumed state and acquisition conditions, the observable fields expected from the tool, and the propositions those fields could change. These records separate action selection from retrieval ranking and prevent an unrelated observation from triggering a global revision. Section 6.1 compares the design with existing workflows and agent architectures.

The tools provide complementary evidence. DNS rechecks establish resolution under a stated resolver and time; redirect chains expose endpoints and distribution identifiers; APK comparisons reveal content reuse when samples remain accessible. The framework connects these observation types to competing explanations and selects the action with the greatest distinguishing value per assigned cost. Retrieval, execution, parsing and conditional action ranking are exercised in the evaluation.

Consider an inactive entry whose historical redirect infrastructure is still observable. The open question is whether a new entry links to the historical content. The planner compares candidate tools by the explanations their outcomes would separate; execution retains call parameters, original-result locations and acquisition times. Only actual returns become evidence. Expectations stay predictions.

In Figure 2, URL identifiers, historical records and unresolved gaps define a retrieval query q_r. BM25 and dense encoding generate lexical and semantic candidates, RRF combines their rankings, and a cross-encoder reranks query–document pairs. Graph2hop provides a targeted relation path, while CE+Graph appends graph candidates to the reranked list. Source, time, acquisition conditions and original content are reviewed before a retrieved item enters B(k). Retrieval supplies candidate evidence; resolution, distribution continuity and attribution are determined from the corresponding DNS, redirect and content observations.

The system exposes collection, extraction, relation-query and sample-parsing functions through standard tool interfaces. Two retained MCP-facade snapshots cover candidates, neighbourhoods, clusters, probes and controlled submissions. Embedding and reranking provide the retrieval path, while the replay adds scenario-pair discrimination, cost-aware selection and shared scenario adjudication.

3.2 Object Changes and Tool Observations

Let τ mark the time of the external object and k the verification round. The state X(τ) collects access, infrastructure, content and sample features; u(τ) covers object adjustments or environmental events; C(τ) records resolver, access method, collection time, source and completeness. An action q(k) returns an observation o(k+1), as Equation (1) states.

$$\begin{aligned} X(\tau + \Delta) &= T[X(\tau), u(\tau), \varepsilon] \\ o(k+1) &= G[X(\tau), C(\tau), q(k), \eta] \end{aligned} \tag{1}$$

Table 1 Method components and their roles in this study

Methodological basis	Question examined	Use and implementation status
Multisource collection and relation graphs	Which objects can be verified?	The existing system supplies candidates and technical relations
Retrieval augmentation and reranking	Which materials enter the result window?	Existing components are evaluated in six retrieval configurations
Tool agents	How are tools called and responses retained?	Existing orchestration is examined through trace verification
World-model planning	What could the next tool return, and which explanations could it distinguish?	Authored expectations and action selection are implemented in replay
Provenance and proposition management	Which association judgement is affected by a new result?	Cross-layer propositions and a local revision structure are proposed

T describes how the external object changes and G what a tool returns given the state and acquisition conditions; ε and η absorb what we leave unmodelled. Entry migration with retained redirect endpoints and contents is a hypothesis about T. An error under one request condition and substantive content under another is a question about G. A DNS query or an APK read obtains an observation without causing infrastructure migration or modifying the stored resources.

We specify T, Σ and the revision operator and define G through observable tool fields. The 58 retained calls identify which fields each interface exposes, whether a preview is complete and which acquisition conditions accompany the result. The controlled replay uses authored scenario specifications so that every selector is evaluated against the same observable outcomes.

Across the 58 calls, 220 of 292 declared schema-field slots (75.3%) were visible in retained previews; for the eight tools with an independent schema, 157 of 229 slots (68.6%) were visible. This field profile defines the observable schema used to score candidate returns and identifies where complete-result retention is required.

3.3 Cross-Layer Evidence States and Proposition Dependencies

The evidence state B(k) holds observations O(k), relations E(k), propositions H(k) and unresolved gaps U(k), as Equation (2) defines.

$$B(k) = \{O(k), E(k), H(k), U(k)\} \tag{2}$$

Each proposition records a subject, the relation or attribute, its object, the applicable time and conditions, and a judgement status. Evidence fields carry content, source, acquisition activity, validity time, environment and the location of the complete result. E holds support, conflict, dependency and provenance relations. Observed, Inference and Unknown label a direct finding, a derived explanation and insufficient information.

A shared registrar supports a service-level association, an identifier read inside a redirect chain can support a distribution link, and identical resources support reuse. Attributing those objects to one operator takes additional provenance and an examination of alternatives. Separate propositions prevent a technical similarity from becoming an attribution claim, while reposts point back to the original material. Each acquisition retains its own activity and conditions.

URLs and filenames tell us where an object was found; constants, payloads and code can provide evidence of continuity. Figure 3 separates claims about DNS resolution, distribution and reuse for that reason. G constrains interpretation through the acquisition conditions, and the update operator identifies the resolution claim and the dependencies a new return affects. Historical evidence keeps its original time and conditions. A current DNS failure does not revise independently supported distribution or payload claims unless a dependency ties them to the failed resolution; assessing migration needs further observation. h_1–h_3 denote the three propositions shown in

3.4 Candidate Explanations and Expected Action Outcomes

The scenarios in Σ represent migration, termination or platform/template reuse. Each contains B(k), its required conditions, the prediction window Δ, the candidate state, and the supporting and opposing observation sets (Equation (3)).

$$\sigma_j = \{B(k), c_j, \Delta, \hat{X}_j(\tau + \Delta), O_j^+, O_j^-\} \tag{3}$$

Once an entry stops resolving, a complete redirect chain that carries a historical distribution identifier and

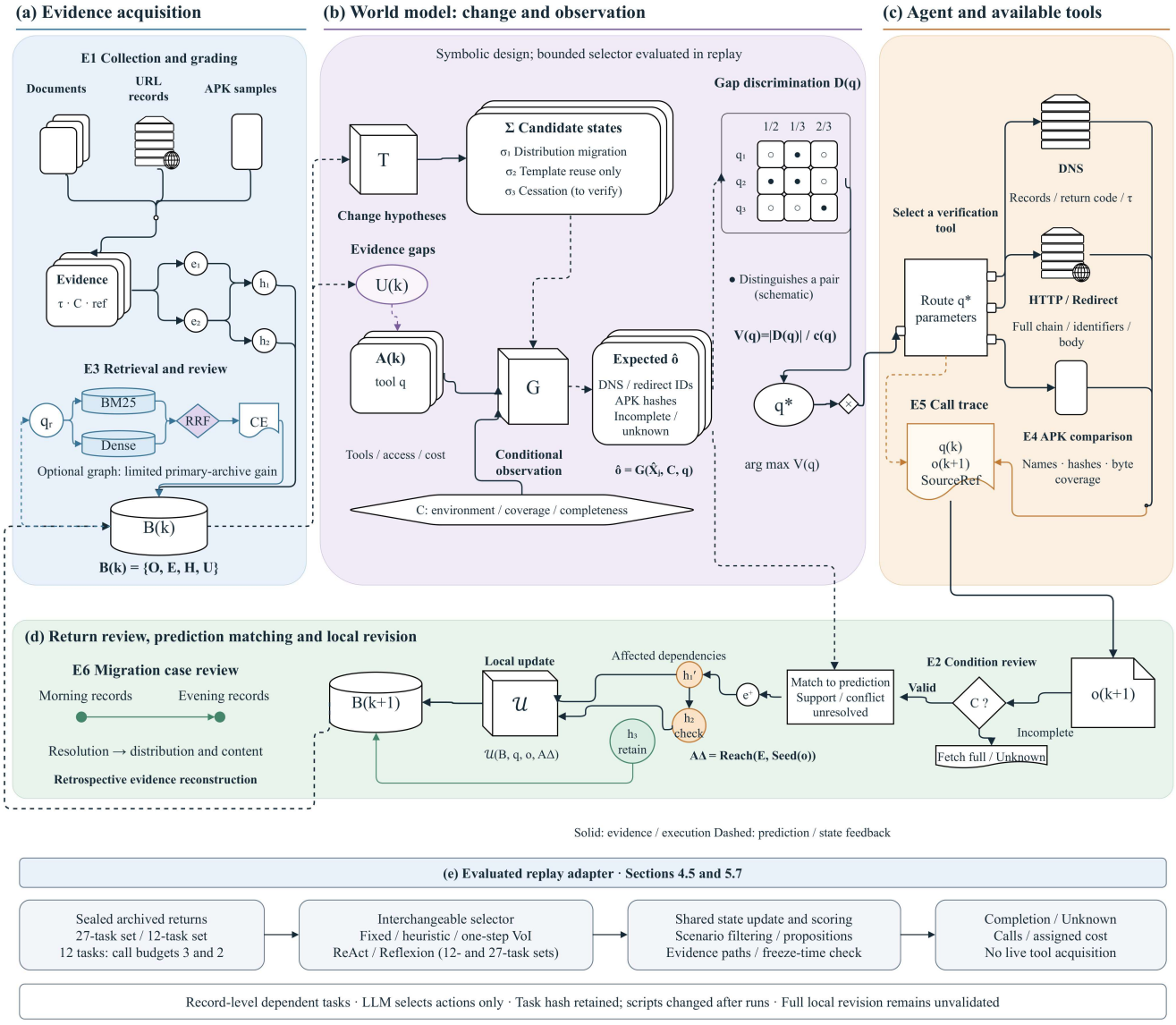


Fig. 1 Symbolic world-model planning and tool execution for fraud-related website lead mining. B(k) contains observations, evidence relations, propositions and gaps. T represents candidate external changes; Σ contains competing explanations; G predicts observable DNS, redirect and APK fields under conditions C. The D(q) matrix records the scenario pairs distinguished by each action, and q* is the selected action. Returned observations pass through condition review before proposition-scoped revision. Solid lines denote evidence or execution; dashed links denote predictions, state feedback or call-record associations. Panels (a)–(d) show the complete framework, and panel (e) shows the shared replay interface used to compare action selectors

the corresponding content may support migration, while a shared registrar or a generic template is also consistent with reuse. Before treating the absence of historical content as evidence against migration, the investigator has to check whether request conditions and collection coverage could have exposed it. That check is necessary, though not sufficient, for reading a negative return.

Expected outcomes are expressed in fields the tool can observe: DNS record types, addresses and validity times; redirect steps, endpoints and identifier context; sample-entry matches, hashes and byte volumes; publication times and provenance. The outcome set includes unmet conditions, truncated results and indeterminate returns. It is a finite catalogue of field-level responses used for discrimination rather than a probabilistic belief distribution.

3.5 Expected Distinguishability and Action Selection

For a set of comparable scenarios Σ, D(q) is the set of scenario pairs that the observable outcomes of action q can distinguish. We use unweighted discrimination coverage:

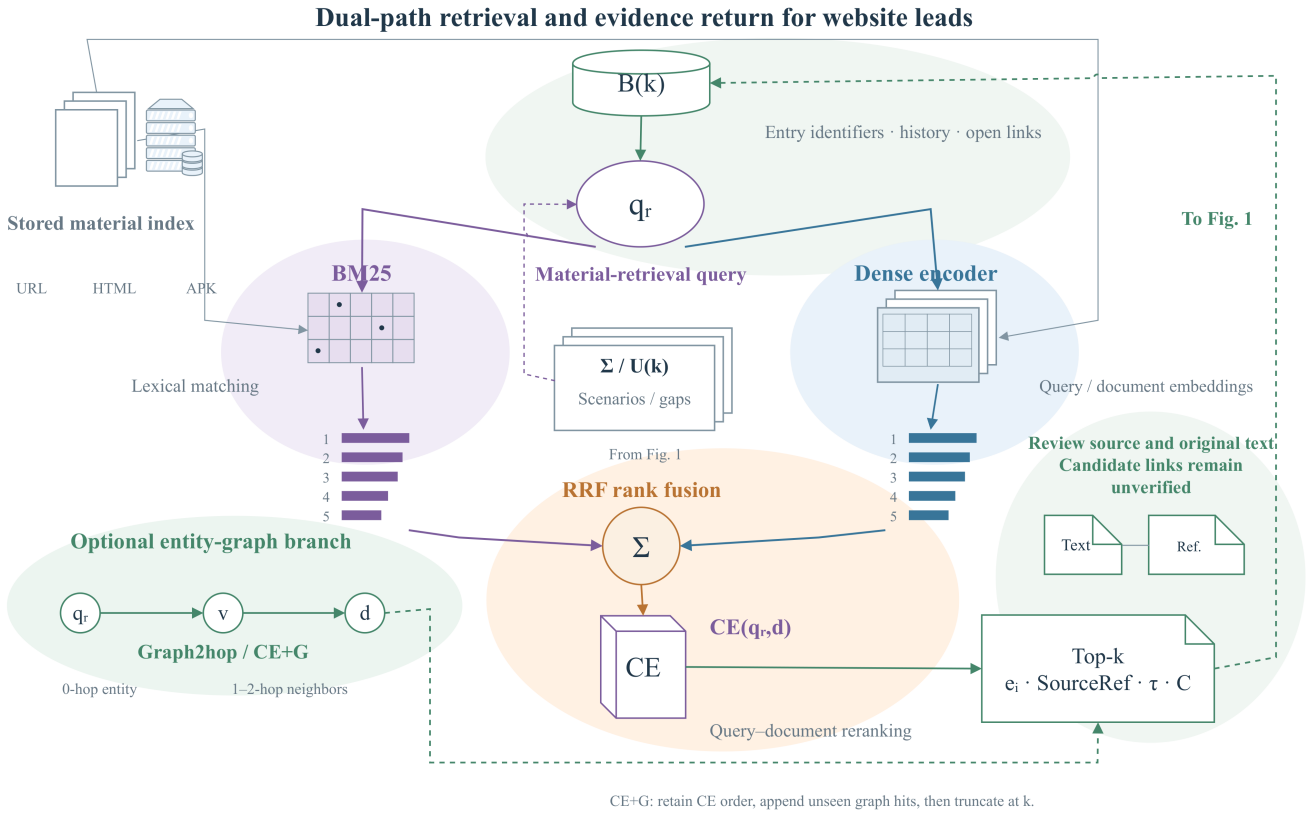


Fig. 2 Evidence retrieval and relation organisation for website-lead mining. Retrieval query q_r differs from verification action q(k). The scenarios Σ and gaps U(k) in Figure 1 guide query formation. BM25 and dense paths converge through RRF and CE; the optional graph branch appears at lower left. Rank bars indicate order rather than scores. Peripheral query, review and return interfaces show the proposed connections. Green dashed links denote the optional branch or return. DNS, HTTP and APK observations enter through the execution stage in Figure 1

Table 2 Key fields in state records

Record type	Key fields	Analytical role
Object	Anonymous ID, layer, current and historical attributes	Distinguish access entries, infrastructure and content
Observation	Result, source, acquisition time, validity time, conditions	Assess whether a result applies to the current question
Proposition	Subject, relation, object, conditions, status	Limit the interpretation of a technical association
Evidence relation	Support, conflict, dependency, provenance derivation	Locate the revision scope and identify duplicate sources
Candidate outcome	Explanatory premises, time window, expected and counter-observations	Specify the basis of pre-action simulation
Verification record	Candidate tools, parameters, expectations, actual results, state versions	Compare selection grounds with execution feedback

$$V(q) = \frac{|D(q)|}{c(q)}$$
$$q(k) = \underset{q \in A(k)}{argmax} V(q) \tag{4}$$

Permissions, availability, budget and a cost c(q) recorded before selection constrain the feasible set A(k), and Equation (4) defines the selection criterion. Applying it across tools requires consistent definitions of cost and outcome partitions; unweighted coverage can favour a tool whose distinguishing outcomes are rare. Because the operational logs use heterogeneous cost measurements and do not contain pre-action scores, Section 5.7 evaluates the criterion with explicit replay cost tables. The criterion resembles Bayesian optimal experimental design, which selects observations by expected information gain [33].

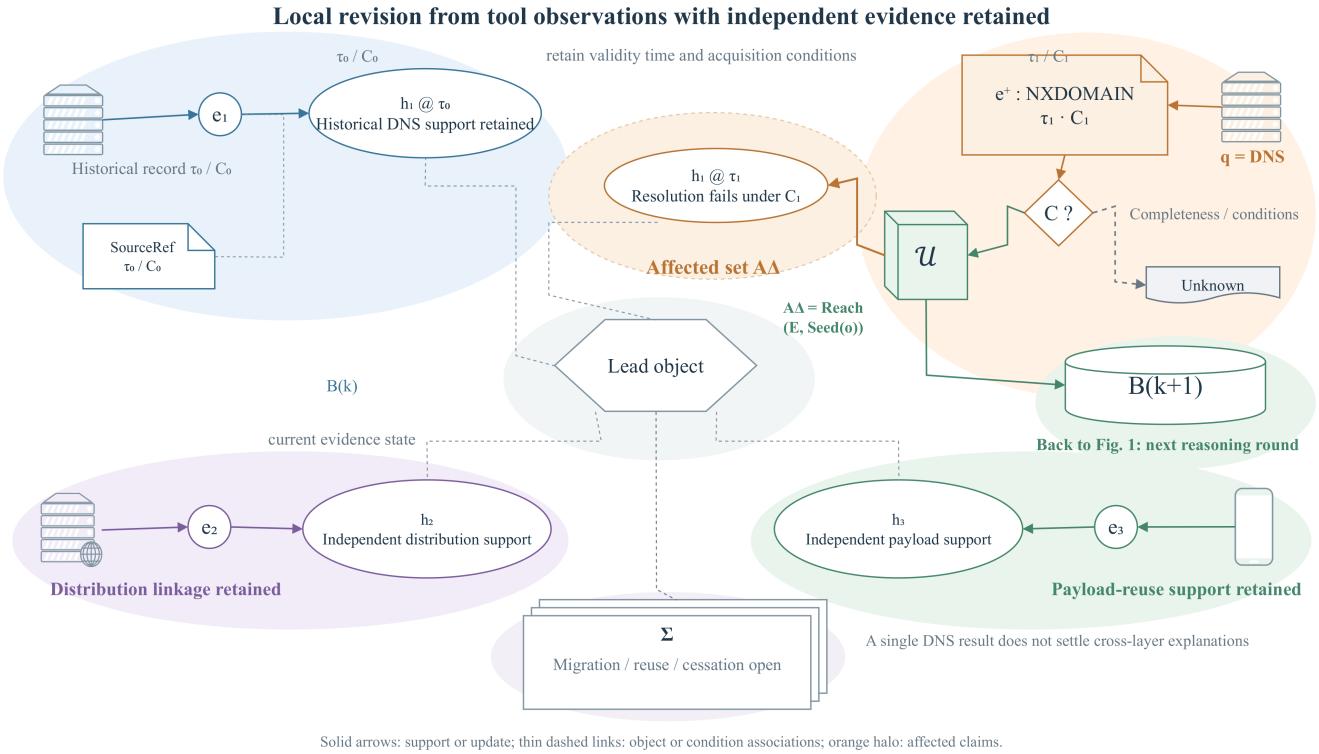


Fig. 3 Figure 3. Layered interpretation of tool observations and dependency-scoped revision. The DNS return passes through condition review before it is linked to evidence and claims. The orange halo marks the current resolution claim. Historical DNS, distribution and payload evidence occupy separate branches. NXDOMAIN at τ_1/C_1 establishes neither full inaccessibility nor cessation. $h_1@\tau_0$ retains historical support; h_2/h_3 retain independent support in the absence of propagating dependencies. Σ remains unresolved. Thin dashed links show object or condition associations, not dependencies or source independence. τ_0/C_0 and τ_1/C_1 distinguish prior and current observations. Table 2 lists the state-record fields used in the subsequent update steps

The replay makes D(q) computable from the retained scenario expectation table. For each unordered pair of surviving scenarios, it compares the predicates for every field marked both visible and eliminative for action q, and counts the pair once if at least one such field distinguishes the scenarios. Missing or unconstrained expectations do not contribute. The assigned cost c(q) is read from the common cost table before selection. The selector maximises $|D(q)|/c(q)$, breaks ties by larger $|D(q)|$, then by lower cost and finally by action identifier, and returns Unknown if every remaining score is zero.

After each return, fields marked as both visible and eliminative are checked against the scenario expectations. Scenarios contradicted by those fields are removed, and the scores are recalculated over the surviving scenarios. Missing or unconstrained fields do not eliminate a scenario. If a return contradicts every scenario, the implementation restores the previous set and records the contradiction. The loop stops when one scenario remains, the budget or deadline is reached, or no remaining action has a positive score; propositions not determined at stopping remain Unknown.

Task V01 illustrates the selection rule. Its three scenarios encode a matching registration handle with current resolution, the same handle without current resolution and a different handle with current resolution. Graph lookup distinguishes two scenario pairs at an assigned cost of 1 s, while DNS distinguishes two pairs at 5 s. The rule therefore selects graph lookup first; after the positive graph return eliminates the third scenario, DNS separates the remaining pair. The resulting sequence resolves V01 in two calls and 6 assigned seconds.

This construction relates to belief-state action selection in partially observable decision processes [17]. B(k) is a symbolic evidence state, and Equation (4) implements one-step, cost-normalised discrimination over that state, consistent with information-value analysis [15]. The competing scenarios make alternative explanations explicit, as in Analysis of Competing Hypotheses [13]. Equation (5) draws on truth-maintenance systems [7] to limit revision to propositions connected to the new observation.

For a question about historical distribution continuity, a DNS recheck clarifies resolution under the specified conditions, a complete redirect trace exposes identifiers

in context, and an APK comparison tests resource reuse when a sample remains accessible. Retrieval and graph paths locate the relevant materials. Their results still need source, time and full-content review before they support an association claim.

3.6 Execution Feedback and Local Revision

A new observation identifies the propositions whose object, time and attributes intersect, and dependency traversal gives the set AΔ to re-evaluate (Equation (5)).

$$\begin{aligned} A_{\Delta} &= Reach\,(E, Seed(o)) \\ B(k+1) &= \mathcal{U}\,[B(k), q(k), o(k+1)\ \ , A_{\Delta}] \end{aligned} \tag{5}$$

Seed(o) holds the propositions that a new observation's conditions directly support, contradict or affect, and Reach follows their dependencies. The revision record keeps the expectation, the actual result, and whether the comparison supports, conflicts with or leaves a proposition unresolved. This is an evidential comparison rather than an arithmetic update. Truncated previews require the complete result, and conflicting returns require an examination of object changes and acquisition conditions; independently supported claims are retained. Recent work examines whether language models revise beliefs coherently when new evidence arrives [14].

A new distribution chain, for instance, can revise the link between a new and an old entry without restoring the old entry's reachability or invalidating the resource-reuse evidence; each judgement retains its own evidence scope. Verification ends when the main gap is resolved, no further tool is available or the budget is spent. The final record states the supported findings and the questions that remain open.

4 Materials and Methods

4.1 Data Sources and Snapshots

We drew on records from earlier phases of the project. The historical collection snapshot uses separately archived query outputs dated 2 October 2026; the retrieval analysis uses per-query evaluations archived on 18 September 2026; log analysis covers JSONL traces from 1 to 8 October 2026 and the corresponding operational session. We compared three Android package builds archived in September 2026 and reconstructed the migration case from the morning and evening records of 8 October 2026, with further calculations on 10 October 2026. Table 3 names the question each analysis addresses. The bounded replay appears in Figure 1(e) and is evaluated separately from E1–E6 in Section 5.7.

We analysed each source by snapshot and version. Archived inputs carry SHA256 records, and the scripts, configurations and anonymised aggregate tables preserve the file correspondence; figures state their objects, times and denominators. Before analysis we replaced all infrastructure identifiers — domains, IP addresses, account handles, session identifiers and package names — with stable pseudonyms. The mapping sits with the case file and can be disclosed to the editors on request.

Research ethics. The 191 candidate addresses in Section 5.6 came from a passive reverse-DNS lookup of two jump-host addresses already present in the case archive. The 188 tested addresses were screened for the case's own page signatures; the retained scan record holds the domain, a signature score and the matched signature names, and no personal data. Page captures are retained as evidence. The study involves no human participants; the two annotators in Section 4.2 are authors, so the institutional review board statement is ' Not applicable'.

Table 3 maps each analysis onto Figure 1(a–d). E1 and E3 assess how evidence forms and is retrieved in panel (a); E2 examines the acquisition conditions used by G and by the return-review gate; E4 tests the content comparisons the tool interfaces allow, and E5 checks call–return correspondence and retention in panel (c); E6 reconstructs the changes in U(k) and B(k+1) in panel (d). E3 alone gives a six-configuration quantitative comparison; the other analyses address collection, observation quality, content overlap and retrospective interpretation.

4.2 Retrieval Metrics and Paired Comparisons

The six configurations are BM25, dense-vector retrieval, RRF, RRF with cross-encoder reranking (CE), independent two-hop graph retrieval (Graph2hop), and a combination of reranked and graph results. The evaluation contains 87 queries and 147 indexed documents after removing queries whose relevant records fell outside the index. The primary relevance rule marks a document when its body contains the target entity; an independently adjudicated 150-pair annotation study assesses that rule.

Label provenance. The reference labels used for the retrieval evaluation came from a rule-based annotation pipeline that applied the written rubric block by block. Two of the authors then labelled all 150 query–document pairs independently, blind to the archived labels and to each other's labels. Every agreement coefficient reported here therefore measures agreement

Table 3 Experimental objects and evaluation questions

Evidence ID and analysis	Archived sample	Framework link and readout
E1 Collection and grading	2 October archive; 10 October review	B(k): candidate consolidation and grading; Section 5.1
E2 Observation conditions	Relation filtering and paired returns from one site	C and G: conditions affecting usable observations; Section 5.2
E3 Retrieval components	87 queries; six configurations	$q_r \rightarrow$ B(k): recall, ranking and paired differences; Section 5.3, Figure 4; Supplementary Figure S1
E4 APK content comparison	Three builds; 6,218, 6,218 and 6,339 entries	o(k+1) $\rightarrow$ H(k): name/content matches and byte coverage; Section 5.4
E5 Agent traces	153 events; 32 sessions; one matched operational archive	q(k) $\rightarrow$ o(k+1): call provenance and retained returns; Section 5.5
E6 Migration case	Morning and evening records on 8 October	U(k) and B(k+1): retrospective changes in specific claims; Section 5.6

between the two annotators, or between their adjudicated labels and the archived pipeline.

Annotation reliability. Two authors independently labelled all 150 query–document pairs, blind to the archived labels and to each other's labels. Agreement was $\kappa = 0.79$ on a 60-pair stratified subsample (95% CI 0.59–0.96) and $\kappa = 0.89$ on the full pool (95% CI 0.80–0.97); six disagreements were resolved by the two annotators. The entity-presence rule retained every archive-positive pair and therefore serves as a high-recall evaluation rule. Full comparisons among the archived, adjudicated and rule-based labels are reported in Supplementary Tables S1–S4.

For retrieval evaluation, z indexes a retrieval query q_r. Recall@K measures how many of a query's relevant documents appear among the first K results, and mean reciprocal rank (MRR) measures where the first relevant result appears:

$$\begin{aligned} Recall@K(z) &= \frac{|L(K,z) \cap Y(z)|}{|Y(z)|} \\ MRR &= \frac{1}{N} \sum_z RR(z) \end{aligned} \tag{6}$$

RR(z) is the reciprocal rank of the first relevant result, or zero when no relevant result is retrieved; L(K,z) is the top-K result set and Y(z) the relevant set in Equation (6), with N the number of queries. We compare configurations on paired queries, report mean paired differences and win/tie/loss counts, draw 10,000 paired bootstrap samples with seed 20261010, apply two-sided exact sign tests after excluding ties and use Holm correction across all 12 contrasts. Reported intervals are percentile intervals and reported p-values are multiplicity-adjusted.

4.3 Client-Content Comparison

The client experiment separates filenames from file contents across three builds from one case archive. build-A and build-B each contain 6,218 entries; 6,212 entries (99.90%) have identical counterparts, including 6,154 pairs with different filenames. Shared content accounts for 18.354% and 18.355% of uncompressed bytes, localising most byte-level change to a small set of large entries. build-C contains 6,339 entries, of which 6,185 match content in build-A. Entry-level pairing therefore identifies content preserved across renamed packaging and complements code- and structure-level repackaging analysis. Two unrelated packages provide negative controls at 2.44% and 2.77% entry overlap.

Entry counts, distinct content-hash counts and uncompressed byte volume describe payload units, content diversity and matched volume; together they separate the contribution of many small resources from that of a few large files. Prior work detects Android repackaging from layout fingerprints [25] and classifies malware from application features [1].

4.4 Agent Traces and Case Analysis

We classified sessions into four provenance categories: an explicit test identifier, an exact match to test source code, a match to the operational JSON archive, or unconfirmed provenance. For traceability and correct pairing we checked tool names, parameters and response previews, examined whether previews were parseable and whether they reached the 2,000-character retention limit, and for the operational session compared call order, parameters and parseable fields with the archived record.

The case analysis reconstructs how later evidence changes proposition-level state in a completed investigation. We identify what was observable at each stage, formulate competing explanations and trace which propositions each subsequent observation affected.

4.5 Sealed-Return Replay Design and Information Boundaries

We evaluate action selection in three stages. A three-task demonstration establishes the executable selection loop. Two sealed-return sets then compare fixed, heuristic and one-step rules under identical archived returns and action costs: 27 two-scenario tasks from three case archives and 12 three-scenario tasks from one domain ledger. The 12-task set is evaluated at two- and three-call budgets, and a separate APK-cost sensitivity analysis re-costs the recorded sequences.

Completion is scored separately from scenario convergence. A task is complete only if every non-excluded proposition matches its reference label, each asserted proposition cites an evidence path that resolves to an existing file, and those supporting files pass the freeze-time check. The time check uses file modification times with the configured tolerance. These checks establish agreement and file provenance within the replay; they do not independently validate the semantic sufficiency of every cited item or establish field-investigation accuracy. The 27-task set contains 18 tasks from 8ssm, seven from okaxpxap and two from 440me. All 12 tasks in the other set derive from the 8ssm ledger; its two budget settings are not independent samples.

The replay module separates task loading, action selection, release of the selected return, shared scenario filtering and final scoring. ReAct and Reflexion replace the action selector while the replay holds scenario filtering, proposition judgement and evidence assembly constant. Completion therefore measures action-selection performance under a shared adjudication layer. The nospec ablation removes probe descriptions and return-field names while leaving action identifiers visible.

For the sealed sets, completion requires every scored proposition to match the task answer and every assertion to resolve to a pre-freeze evidence file. We report calls, assigned probe cost, empty returns and Unknown propositions alongside completion. Final completion and scenario convergence are evaluated separately from the loop's stopping label, so a decisive final call is credited even when the budget is exhausted.

The sealed-return protocol fixes the observations available to every selector. Task hashes identify retained inputs, and Supplementary Appendix B records the task sets, scoring conditions and implementation-version provenance.

5 Results

5.1 Collection Consolidation and Grading (E1)

The historical snapshot holds 772 document records. Of these, 759 have nonempty body text and normal status, and 13 record collection failures. From 162,458 candidate occurrences we obtained 2,476 distinct values, which form 2,476 lead-pool records — a consolidation ratio of 65.61:1. The pool holds 28 high-, 1,783 medium-, 581 low- and 84 discarded-grade records, and the counts sum to the pool total. The read-only review snapshot holds 796 document records, 167,250 candidate occurrences and 2,589 lead-pool records.

Grading ranks candidate observations for B(k), and source and condition checks then decide whether they can support propositions in H(k). Of the 28 historical high-grade records, 11 passed both checks used at the time; the other 17 did not satisfy the release prerequisites. A high extraction grade therefore marks a candidate for further checking, not an association judgement independent of provenance, time or relation specificity.

5.2 Effects of the Observation Environment on State Interpretation (E2)

Two related tables differ by more than their size: reading an internal proxy address as a hosting location would create a spurious shared-infrastructure association. The current relation table and the cleaned view hold 2,693 and 2,430 edges over 16 relation types; applying the project's IP-exclusion configuration cut the counts to 2,182 and 1,926. The excluded records include documented internal proxy-address families and invalid addresses. Observation-environment checks therefore change the usable evidence set, not just its size.

These records show why G has to be conditioned on C. Where access or completeness falls short, failure to obtain body text leaves a gap in U(k). An absent identifier can likewise weaken a proposition in H(k) that predicts its presence — provided the acquisition conditions would have made it observable.

One historical record shows how tightly request conditions and returned content are coupled. Under the original access conditions, the request returned HTTP status 400 with a 307-byte body; adjusting the recorded request conditions returned 1,911 bytes of content, with the same length in five repetitions.

The cleaned view contains 2,430 archived relation records before condition screening. These technical links locate candidate sources for B(k), but an observation can support a specified proposition in H(k) only after source, time and acquisition checks. Proposition dependencies

in E(k) must be recorded separately, because an object-level graph edge does not define one.

5.3 Evidence Acquisition by Retrieval Components (E3)

Cross-encoder reranking gave the clearest gain in early retrieval (Table 4; Figure 4). Recall@5 rose from 0.7671 for BM25 and 0.6841 for RRF to 0.8868 for RRF+CE, while Graph2hop reached 0.8879 and CE+Graph 0.8881. RRF+CE therefore gained 11.97 and 20.27 percentage points over BM25 and RRF. This gain supplies the evidence-acquisition branch of Figure 1(a); source review determines how retrieved material enters B(k). Figure 4 shows the paired effects, and Supplementary Figure S1 reports the query-level differences.

Cross-encoder reranking is the decisive retrieval component. RRF+CE reaches Recall@5 = 0.8868, compared with 0.7671 for BM25, 0.6841 for unreranked RRF and 0.5594 for the dense channel. RRF+CE versus RRF isolates the contribution of reranking to the fused candidate list, while RRF+CE versus BM25 evaluates the complete retrieval pipeline.

Paired analysis puts the mean Recall@5 gain for RRF+CE over BM25 at 11.97 percentage points, with a 95% resampling interval of [7.20, 17.15]; 24 queries improved, 61 were unchanged and two declined. Against RRF, Recall@10 rose by 13.46 points with an interval of [9.00, 18.30]. Reranking therefore improves the accessibility of early-ranked evidence in this archived task. Supplementary Figure S1 shows the query-level paired differences and retains the heterogeneity that the macro-averages hide.

Table 5 reports four paired component contrasts across Recall@5, Recall@10 and MRR: RRF+CE against RRF isolates adding reranking to the fused candidate list; RRF+CE against BM25 compares complete retrieval configurations; Graph2hop against RRF+CE compares alternative acquisition routes; and CE+Graph against RRF+CE measures what appending graph candidates adds. These contrasts attribute the observed gain to a specific part of the evidence-acquisition branch.

Differences are candidate minus baseline on the same 87 queries. Recall differences and intervals are in percentage points, while MRR differences keep the metric's own scale. Intervals are 95% paired-bootstrap percentile intervals, and adjusted p-values come from two-sided exact sign tests that exclude ties, with Holm correction across all 12 contrasts. Table 5 adds the archived MRR results to the earlier recall comparisons and uses no additional retrieval runs.

RRF+CE raised MRR by 0.0966 over BM25 and 0.1032 over RRF, with Holm-adjusted sign-test p-values of 0.00415 and 0.00130. All six reranking contrasts had adjusted p-values below 0.05. CE+Graph remained within one percentage point of RRF+CE on Recall@5, while Graph2hop produced query-specific gains and losses around the reranked stack. CE reranking is therefore the default retrieval path; graph retrieval is reserved for relation-sensitive queries.

RRF+CE and Graph2hop scored identically on Recall@5 for 80 queries, with Graph2hop better on four and worse on three. That variation raises the question the world-model formulation asks before any tool is called: which relation is this query missing? U(k), Σ and the expected returns of G make that question explicit.

The graph configurations expose relation-linked candidates for a small subset of queries while preserving the aggregate performance of the reranked stack. Graph2hop changes Recall@5 on seven queries and MRR on five; CE+Graph changes recall on one query. The result supports using graph retrieval selectively when the missing evidence is relational, with cross-encoder reranking as the default path.

5.3.1 Where the Gain Comes From, and What the Ceilings Mean

The aggregate figures hide a strong ceiling. BM25 already reaches Recall@5 = 1 on 48.3% of the 87 queries and Recall@10 = 1 on 82.8%; under RRF+CE those shares rise to 63.2% and 92.0%. Half of the query set therefore has no room to improve on Recall@5, and almost none on Recall@10. Sorting queries by their BM25 score shows where the paired gain sits: in the lowest quartile (mean 0.333) reranking adds 40.51 percentage points, in the second (0.754) 6.82, and in the two upper quartiles 0.00. Over the 45 queries with BM25 below 1, the summed gain equals 51.39% of the summed remaining headroom.

Reranking delivers its largest benefit where lexical retrieval leaves headroom. The ten queries with the largest improvements account for 62.31% of the summed positive difference against BM25, while leave-one-query-out analysis keeps the mean Recall@5 gain between +10.94 and +12.34 points. Together with the +40.51-point gain in the lowest BM25 quartile, this identifies difficult early-retrieval queries as the main operating region for reranking.

Restricting relevance to a strict rule retained 77 of the 87 queries and preserved the reranking result. On this subset, cross-encoder reranking improves Recall@5 over BM25 by +14.43 percentage points, compared with +13.52 points under the loose labels on the same queries. Configuration-specific graph results under the stricter rule are reported in Supplementary Figure S2.

Table 4 Results of the six retrieval configurations

Method	Recall@5	Recall@10	MRR
BM25	0.7671	0.9263	0.8780
Dense	0.5594	0.6986	0.7424
RRF	0.6841	0.8453	0.8715
RRF+CE	0.8868	0.9799	0.9746
Graph2hop	0.8879	0.9826	0.9828
CE+Graph	0.8881	0.9822	0.9746

Table 5 Paired retrieval comparisons on the same 87 queries

Candidate − baseline	Metric	Mean Δ	95% interval	Win / tie / loss	Holm p
RRF+CE − BM25	Recall@5 (pp)	+11.968	[+7.203, +17.146]	24 / 61 / 2	1.05e-04
RRF+CE − BM25	Recall@10 (pp)	+5.365	[+2.586, +8.509]	13 / 73 / 1	0.0128
RRF+CE − BM25	MRR	+0.0966	[+0.0500, +0.1481]	15 / 71 / 1	0.0042
RRF+CE − RRF	Recall@5 (pp)	+20.267	[+15.374, +25.562]	45 / 42 / 0	6.82e-13
RRF+CE − RRF	Recall@10 (pp)	+13.461	[+9.002, +18.301]	32 / 54 / 1	8.71e-08
RRF+CE − RRF	MRR	+0.1032	[+0.0576, +0.1548]	17 / 69 / 1	0.0013
Graph2hop − RRF+CE	Recall@5 (pp)	+0.109	[-3.129, +3.218]	4 / 80 / 3	1.0000
Graph2hop − RRF+CE	Recall@10 (pp)	+0.268	[-2.797, +2.854]	5 / 81 / 1	1.0000
Graph2hop − RRF+CE	MRR	+0.0081	[-0.0287, +0.0469]	3 / 82 / 2	1.0000
CE+Graph − RRF+CE	Recall@5 (pp)	+0.128	[+0.000, +0.383]	1 / 86 / 0	1.0000
CE+Graph − RRF+CE	Recall@10 (pp)	+0.230	[+0.000, +0.690]	1 / 86 / 0	1.0000
CE+Graph − RRF+CE	MRR	+0.0000	[+0.0000, +0.0000]	0 / 87 / 0	1.0000

Figure 4 shows the paired Recall@5, Recall@10 and MRR differences across all 87 queries in a raincloud plot. Points retain individual queries; distributions, interquartile ranges and medians show the frequency and magnitude of query-level changes. Recall differences use percentage points, whereas MRR retains its original scale. Table 5 provides mean differences, bootstrap intervals and adjusted p-values, and Supplementary Figure S2 compares the archived configuration scores under the two evaluation rules.

5.4 Separating Name Changes from Shared Content (E4)

The two builds look like the same package assembled twice: each archive contains 6,218 ZIP entries with unique names, and the two name sets barely overlap. Of the 62 same-name pairs, 58 are byte-identical and four differ in content; the remaining 6,156 names in each package occur only there. Filename comparison therefore captures packaging-level differences, whereas content comparison asks whether payloads persist.

Content comparison finds what filenames hide. The full-content hash comparison in Supplementary Table S10 finds 6,154 differently named resource pairs with identical uncompressed contents, spanning 6,002 distinct nonempty content hashes; every one of these entries exceeds 64 bytes. Adding the 58 same-name, same-content pairs gives 6,212 shared pairs, or 99.90% of the entries in each package. Only two differently named entries per package stay unmatched.

Among the 6,154 differently named shared resources, the smallest entry is 68 bytes, the median 420 bytes and the largest 774,549 bytes. The 850 entries of at least 1 KiB account for 13.81% of entries and 89.51% of shared-resource bytes, and the 50 entries of at least 100 KiB account for 0.81% of entries and 58.90% of bytes.

Shared contents total 22,052,524 bytes, or 18.354% and 18.355% of uncompressed bytes in the two packages. They comprise 20,807,784 bytes of differently named resources, 1,065,272 bytes of native libraries and 179,468 bytes of other stable same-name entries. The four same-name pairs with different contents account for about 81.167% of uncompressed bytes in each package. Entry counts therefore show broad resource reuse, whereas byte volume localises most change to a few large entries.

The four same-name differences are AndroidManifest.xml, META-INF/MANIFEST.MF, classes.dex and resources.arsc. Together they account for 81.1672% of uncompressed bytes in build-A, localising most byte-level change to bytecode, compiled resources and metadata. Hash comparison establishes entry-level continuity and change; code, configuration and distribution evidence provide the corresponding behavioural and operator-level interpretation.

The content hashes give observations O(k) that can support a resource-reuse proposition in H(k) even when filenames change. Platform or template reuse remains in Σ as an alternative to continuity of the application itself.

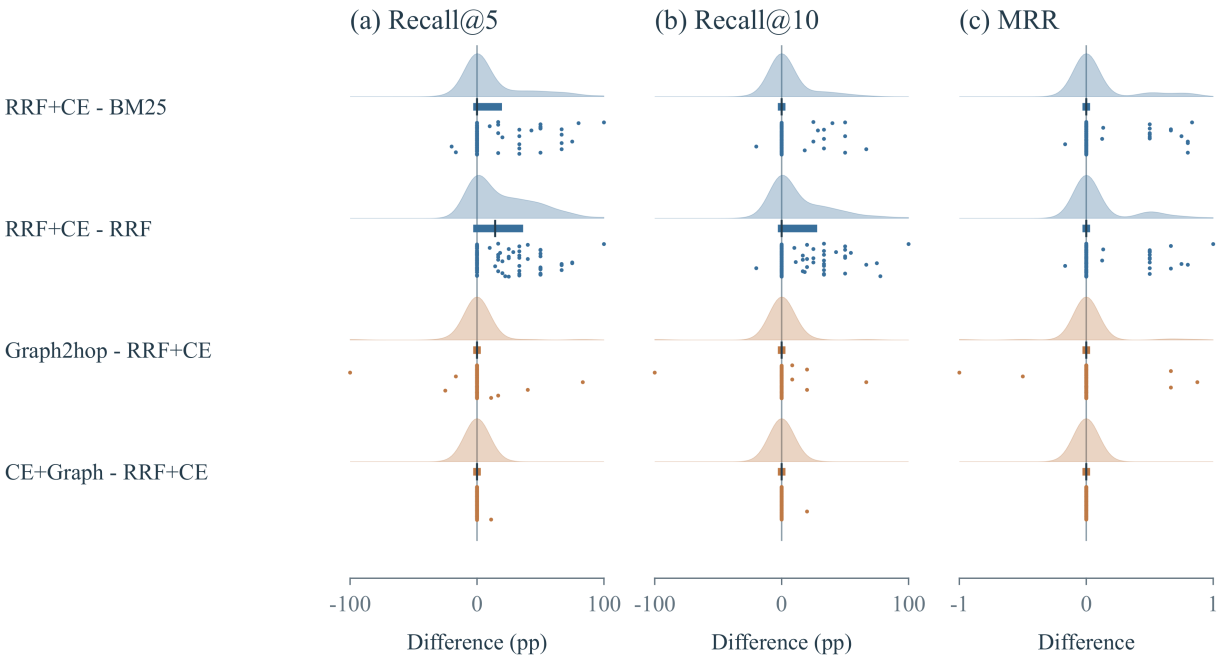


Fig. 4 Paired effects of reranking and graph retrieval. The four contrasts use the same 87 queries; positive differences favour the candidate configuration. Blue denotes reranking contrasts and orange denotes graph contrasts. Each point represents one retained query. Bands show interquartile ranges and dark marks show medians. Half-density curves use fixed smoothing widths of 10 percentage points for recall and 0.10 for MRR, normalised within each row. Vertical jitter separates coincident points and carries no data meaning. Mean differences, 95% bootstrap intervals and multiplicity-adjusted tests are reported in Table 5

Neither shared resources nor whole-package differences resolve operator attribution; distribution chains, code and configuration have to close that gap.

Figure 5 compares shared entry content and byte volume across the three related builds.

5.5 Agent Traces and Evidence Retention (E5)

We found 153 valid events forming 32 sessions, with no invalid lines and no exact duplicates. Eighteen sessions carry explicit test identifiers, eight match the tools, parameters and final text of test source code exactly, one corresponds to the operational JSON archive and five have unconfirmed provenance; the 27 sessions with terminal status succeeded across several provenance categories. Provenance labels distinguish test traces from operational activity.

In the operational session, candidate retrieval and lead-pool retrieval each occur twice and the other tools once. Four response previews reach the retention limit, and the operational JSON keeps result_keys rather than full result bodies, so not every observation can be reconstructed.

Retention policy determines replayability. All 58 calls preserve parameters and result previews; 35 previews remain parseable JSON, while 23 reach the 2,000-character preview limit and identify where full-result retention is required. Approval information records execution gates, and the retained field structure specifies which observations can enter the evidence state.

The operational session makes ten calls across eight tool types and reconstructs tool order, parameters, returned field families and 12.0 s of recorded tool-event intervals. Tool names and parameters match the operational JSON at all ten steps, and the top-level keys of the six parseable previews match its result_keys. The trace therefore provides a concrete action-and-observation record for evidence-state reconstruction.

Figure 6 summarises the observed return conditions and field availability by tool.

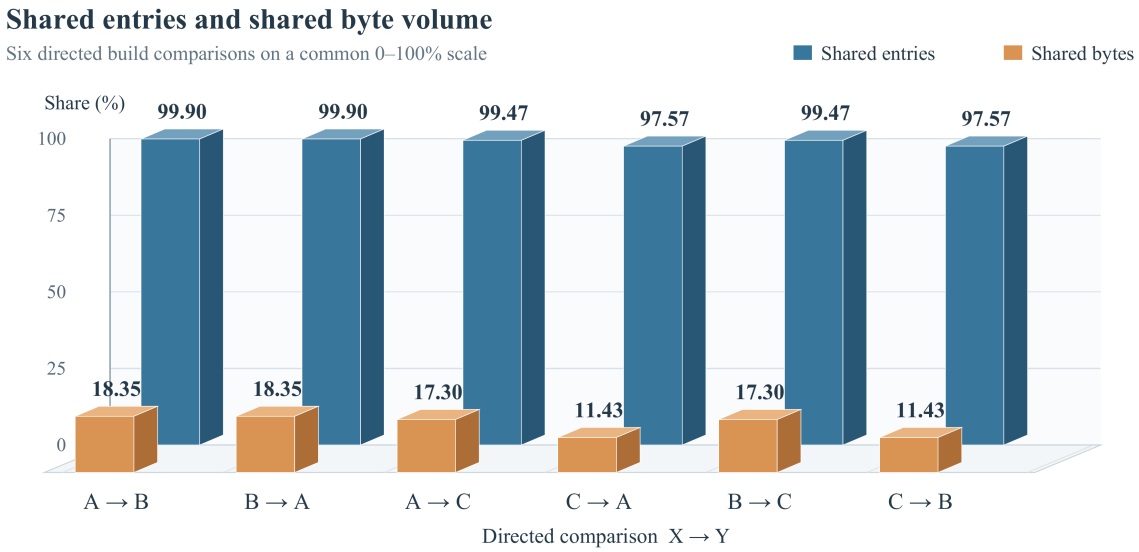


Fig. 5 Shared content across the three related APK builds. The 3D columns compare shared entries (blue) and shared uncompressed bytes (orange) for six directed build comparisons. A, B and C denote build-A, build-B and build-C. For X to Y, build-X is the denominator and shared content occurs in build-Y. Self-comparisons are omitted. Supplementary Table S8 retains the counts and denominators

5.6 Cross-Layer Judgement Changes in the Website Migration Case (E6)

The morning records show the original entry no longer resolving, while the historical redirect infrastructure remained observable; the pages we examined lacked identifiers sufficient to establish distribution continuity. Reconstructing B(k) retrospectively places these findings in O(k) and the entry and infrastructure claims in H(k), each with its own time and conditions. U(k) concerns the link between a possible new entry and the historical distribution, while migration and platform/template reuse remain competing explanations in Σ.

The evening records add a signature-bearing redirect chain from a new entry, historical distribution identifiers and an actual content page. An identifier matching historical material in the new chain can be recorded as Observed; distribution association and entry migration stay Inference, jointly supported by identifier context and chain evidence, with alternatives such as platform or template reuse still open. The resolution result recorded at that time still supports the proposition that the original entry was inactive. The case also retains a pool of 191 candidate addresses, of which we tested 188: eight matched the configured signature and 180 did not. These records describe retrieval and matching outputs alongside changes in the recorded evidence state.

In terms of T, rotating the entry while infrastructure and identifiers persist is a candidate external change, and G describes which chain or content fields expose that change under conditions C. The evening chain narrows the unresolved distribution question and strengthens the corresponding proposition while leaving the old entry's time-specific resolution claim intact. The reconstruction demonstrates proposition-scoped revision.

If the entry rotates again, verification examines integration time, chain endpoints and identifier context. If the identifier disappears while a client remains available, resource, code and configuration comparison addresses continuity at the content layer. The case therefore shows how the framework routes different association questions to different evidence sources.

Table 6 lays out the archived case as a sequence of evidence-state decisions. Each row identifies the available observation, the candidate action, its expected fields and the proposition supported by the returned evidence. The table demonstrates how the framework confines revision to the affected evidence dependencies.

5.7 Frozen Replay of the Planning Layer

Throughout, we use rule for the selection strategy and order for a particular sequence of actions that it produces; the two terms are not interchangeable.

Section 3.5 ranks actions by the scenario pairs they are expected to separate per unit assigned cost, $V(q) = |D(q)|/c(q)$. We first examined this rule in a three-task retrospective demonstration, comparing recorded,

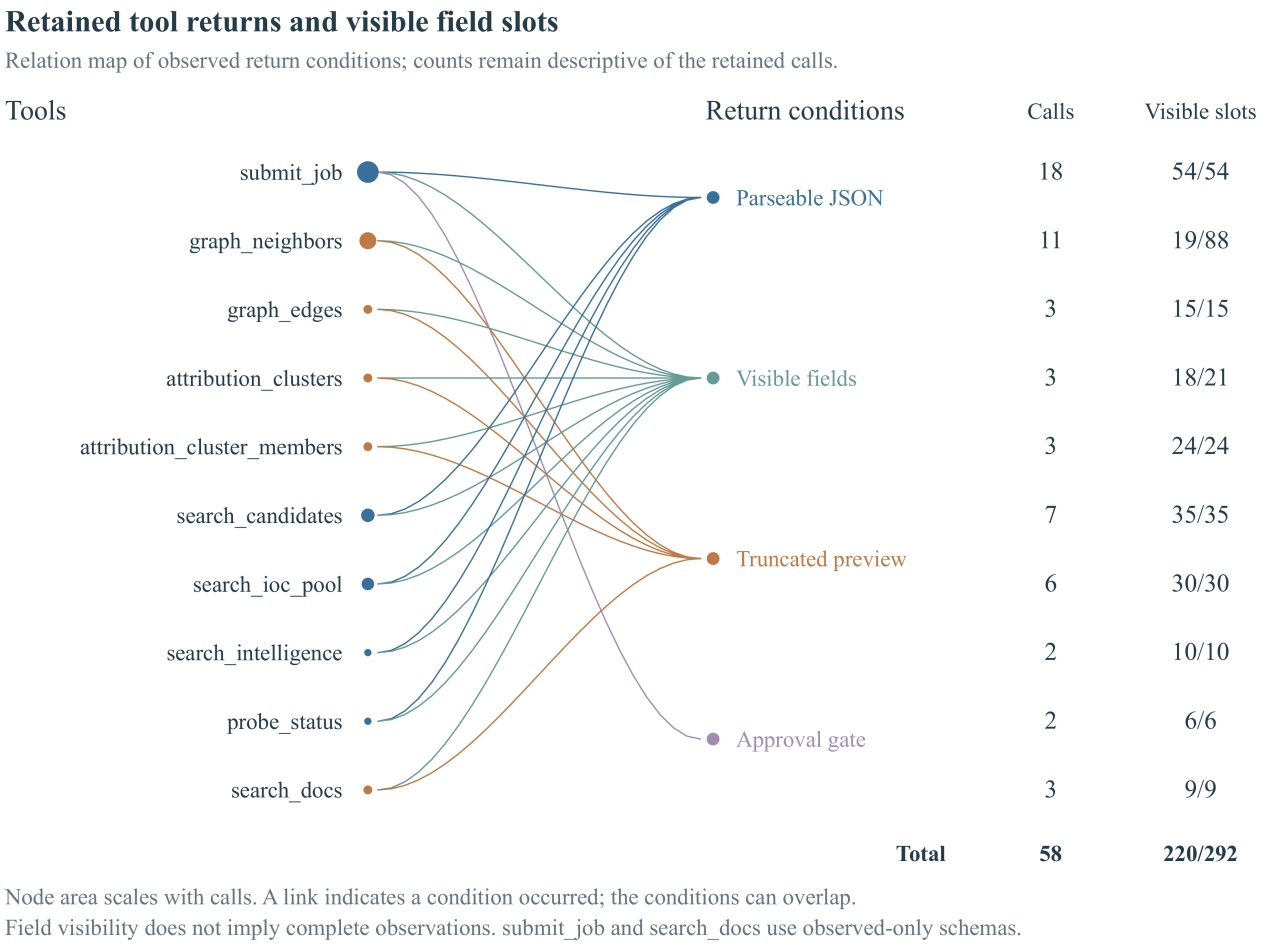


Fig. 6 Retained tool returns and visible field slots. Links connect each tool to conditions observed in its retained returns; conditions may overlap. Tool-node area scales with call count. The numeric columns show calls and visible/declared schema-field slots. The totals are 58 calls and 220/292 visible slots. Truncated-preview and approval-gate categories identify where complete-result retention or execution approval is required

heuristic and one-step rules with ReAct and Reflexion selectors. We then evaluated two sealed-return sets, including a two-budget comparison on the 12-task set. Figure 1(e) shows their shared adapter; Sections 5.7.1–5.7.4 distinguish the task construction and the information each experiment supplies.

Three tasks were replayed (Table 7). T1 asks whether a new entry, entry-X, continues the distribution of an earlier one, entry-W. T2 asks whether entry-Y continues the same operator's landing infrastructure or belongs to a different operator on a shared platform. T3 replays the ten calls of the archived session S01. All three use the same scenarios: migration, platform reuse, and terminated-or-pending. Freeze markers are 00:00 on 8 October 2026 (UTC+8) for T1 and T2 and the session creation time, 00:30:24 UTC, for T3; the starting state was reconstructed as the record then stood.

Four actions were available in every task: graph lookup, DNS recheck, redirect-chain trace and APK comparison. Table 7 reports assigned replay costs derived from paired timestamps and the stated APK default; the 120 s APK sensitivity is reported separately. These costs support within-replay comparison of action sequences under a common cost table.

On T1 all three deterministic orders reached migration, but at very different prices. The recorded order (graph, DNS, chain) used three calls and 61.445 s; the heuristic order (DNS, chain) used two calls and 60.445 s; the rule used two calls and 6.0 s. The rule prefers discrimination per second, so the APK comparison, which should separate migration from platform reuse at 1.0 s, comes first, followed by the DNS recheck that separates the terminated branch at 5.0 s. The chain trace separates the same pair through a token field, but at 55.445 s it is never preferred. T3 behaves differently: the recorded order is ten graph queries, none of which distinguishes the scenarios, because the graph is a static snapshot from 10 September 2026 and returns the same

Table 6 Reconstructed verification loop for the archived migration case

Step	Recorded evidence	Candidate action	Expected return	What the return established
1	Morning record: the entry no longer resolves while the historical redirect chain stays observable	Re-check resolution for a stated time and resolver	Confirmation of non-resolution under C	Supports inactivity for that time only
2	The pages examined carry no distribution identifier	Retrieve historical distribution identifiers from the graph	Identifier set, possibly empty	Empty for the pages examined; the search narrows to infrastructure
3	Evening record: a new entry carries a signature-bearing redirect	Trace the new chain and compare identifier context	Chain endpoints and identifier context	Identifier matches historical material; migration recorded as Inference
4	191 candidate addresses retained in the pool	Probe candidates against the configured signature	Matched addresses	188 tested, 8 matched
5	Content pages recovered from the new entry	Compare against archived content	Payload or resource overlap	Distribution continuity supported; operator attribution left open

edges and empty candidate pools. The task remains unresolved after ten calls and 12.0 s, while the other two orders reach migration in two calls — 60.445 s for the heuristic order and 6.0 s for the rule. On T2 the DNS recheck eliminates migration, because entry-Y resolves to an independent host (198.51.100.77) rather than to the 203.0.113.0/24 jump host. That leaves a pair no action can separate: the chain is observable in one branch only, and no APK sample was obtained through the guard page. The rule stops after one call and 5.0 s with Unknown; the recorded and heuristic orders spend four calls and 62.445 s on the same verdict.

Note. The recorded order is charged from measured event-timestamp differences (its ten calls total 12.0 s), whereas the heuristic and one-step orders are charged from the nominal cost table, which would give 10.0 s for the same ten calls; T1 and T3 share one judgement table, so the three tasks contain two distinct decision problems.

The ReAct and Reflexion rows report means over three repeated runs, with minima and maxima in brackets; each deterministic order has a single sequence. A task is resolved when one scenario remains. Table 7 reports the complete five-order comparison, including the unrounded assigned costs.

Table 7 compares all five orders on the same three archived tasks. Whenever an order reached a verdict, it agreed with the other resolved runs: T1 and the resolved T3 runs converged to migration, and no order resolved T2. Costs differed. ReAct and Reflexion resolved the same sets of runs (3/3 on T1, 0/3 on T2 and 3/3 on T3), but their assigned costs did not. On T1 the mean costs were 60.445 s for ReAct and 61.445 s for Reflexion; on T3 both were 61.445 s. On T2 ReAct used 62.445 s, whereas Reflexion averaged 26.148 s, with calls ranging from 2 to 6 and costs from 6.0 to 62.445 s. At the recorded costs, the rule costs about one tenth of the agent arms on the two decidable tasks (6.0 s against 60.445 s and 61.445 s). Summed over the three tasks, the heuristic order uses 8 calls and 183.335 s, ReAct 9 calls and 184.335 s, and the rule 5 calls and 17.000 s. The difference lies in order: both agents trace the chain early — after the DNS recheck on T1, and immediately after a graph query on two of three T3 runs — whereas the rule selects the 1.0 s comparison first, and the chain accounts for 55.445 s of a 60.445 s sequence. The assigned cost difference therefore mainly reflects action order.

The three-task demonstration establishes an executable selection mechanism and an explicit Unknown outcome. Across the original and sensitivity cost settings, the selector produces inspectable action sequences, exposes the evidence field responsible for each transition and stops when no remaining action distinguishes the scenarios. The sealed-return sets extend this mechanism test under separately specified budgets.

Two features matter for a prospective comparison: the judgement table can be sealed before each return is revealed, and T3 shows what should be recorded, since the session spent all ten calls within one action family and never queried the separating field.

5.7.1 ReAct and Reflexion on the Three-Task Demonstration

For the three-task demonstration in Table 7, both arms ran against the same frozen states, actions and cost table. Each task went to deepseek-chat in three repeated runs, with a cap of six steps. The prompt supplies the task title, the freeze marker, the starting state, the three scenario labels and the action–observation history. After each call, the arm receives the recorded return fields and the scenarios eliminated by that return, using the

Table 7 Frozen replay of three archived tasks under five orders

Task	Order	Calls	Cost (s)	Resolved
T1	Recorded order	3	61.445	yes
T1	Heuristic order	2	60.445	yes
T1	One-step rule	2	6.0	yes
T1	ReAct	2.0 [2, 2]	60.445 [60.445, 60.445]	yes (3/3)
T1	Reflexion	3.0 [2, 4]	61.445 [60.445, 62.445]	yes (3/3)
T2	Recorded order	4	62.445	no
T2	Heuristic order	4	62.445	no
T2	One-step rule	1	5.0	no
T2	ReAct	4.0 [4, 4]	62.445 [62.445, 62.445]	no (0/3)
T2	Reflexion	4.0 [2, 6]	26.148 [6.0, 62.445]	no (0/3)
T3	Recorded order	10	12.0	no
T3	Heuristic order	2	60.445	yes
T3	One-step rule	2	6.0	yes
T3	ReAct	3.0 [3, 3]	61.445 [61.445, 61.445]	yes (3/3)
T3	Reflexion	3.0 [3, 3]	61.445 [61.445, 61.445]	yes (3/3)

same state update as the three deterministic orders. It does not receive the expected-return table, so it cannot compute V(q) for an action it has not taken and must choose from the observations in hand. The asymmetry is deliberate: it asks whether an explicit outcome model adds anything over an agent reasoning from returns. Those return fields are themselves modelled from the expected-return table, not logged, so the arms compare selection rules on a shared reconstructed observation table.

ReAct selects one action or stops at each step. Reflexion runs the same loop first and, if that run does not converge, receives its own trace, is asked what it missed and re-plans once before executing again; reflection is a planning step, not a charged call. Because reflection is entered only after a failed first pass, the T1 and T3 figures describe first-pass runs, while T2 exercises it in all three runs.

ReAct resolved 3/3 runs on T1 and T3 (2.0 calls, 60.445 s; 3.0 calls, 61.445 s) and none on T2 (4.0 calls, 62.445 s). Reflexion matched that pattern — 3/3, 0/3 and 3/3 — at 3.0 calls [2, 4] and 61.445 s [60.445, 62.445] on T1, 4.0 calls [2, 6] and 26.148 s [6.0, 62.445] on T2, and 3.0 calls and 61.445 s on T3. On T2 the reflection round changed neither the verdict nor the stability of the order: the three runs span two to six calls, one ends on an unparsable reply and another at the step cap. Neither arm resolves T2; their results differ in dispersion but share the same unresolved outcome.

The agent comparison uses a shared prompt, task interface and adjudication layer. Model configuration, prompt disclosure and cost accounting are reported in Supplementary Appendix B and summarised in Section 6.4.

5.7.2 Two Sealed-Return Task Sets and Budget Sensitivity

In the 27-task set, the selector resolved 25 tasks in 25 calls and returned Unknown on the two tasks for which no remaining action separated the scenarios. The fixed and heuristic orders completed all 27 tasks using 42 and 40 calls. Assigned costs were 484.560 s, 860.230 s and 566.005 s for the selector, fixed order and heuristic order, respectively. This set evaluates call economy, protocol execution and explicit abstention under identical returns.

The 12-task set tests how action ordering allocates two- and three-call budgets. At two calls, the fixed and heuristic orders complete 4/12 and 3/12 tasks, while the selector completes 12/12. The fixed order spends its budget on DNS and chain tracing; the heuristic order spends it on graph lookup and APK comparison. The selector begins with graph lookup and uses the returned evidence to choose DNS when a second call is needed, resolving all twelve tasks. At three calls all three orders complete 12/12, but the selector uses 21 calls compared with 28 and 30. The result shows that evidence-layer ordering is decisive when the budget binds.

Static-order control. A fixed order (graph, DNS, chain, then APK) reproduces the selector's 12-task outcomes at both budgets. This control shows that the measurable replay gain comes from prioritising the evidence layer that can separate the remaining scenarios within the available budget. The framework's contribution is to make that ordering criterion explicit, inspectable and reusable rather than embedding it in a case-specific fixed workflow. On the 27-task set, the selector uses 25 calls and 484.560 assigned seconds, while the static graph-first order uses 48 calls and 807.785 seconds.

5.7.3 LLM Selectors and Probe-Description Ablation

On the same 12 tasks, ReAct and Reflexion each completed 36/36 runs at each budget: three repetitions per task, arm and budget (Table S20). Both match the one-step rule's task-level sequences and outcomes. Median calls and cost are 2 and 6.0 s; means are 1.750 and 4.750 s. Nine tasks use graph lookup followed by DNS, while three use graph lookup alone, so neither every task nor every run costs 6.0 s. The hosted model is deepseek-chat at temperature 0.7 and max_tokens 320, using prompt version llm_arm_prompt_v1. The labels r0–r2 identify repetitions, not provider-controlled random seeds.

Across 144 full-prompt runs on the 12-task set, ReAct and Reflexion reproduce the explicit selector's task-level sequences and outcomes under both call budgets. Removing probe descriptions and return-field names increases mean calls from 1.750 to 1.944 and assigned cost from 4.750 to 7.235 s, showing that observable field descriptions help the LLM selectors choose the relevant evidence layer.

Across the four full-prompt arm–budget configurations and the ablation, the deepseek-chat archive contains 180 runs and 322 model calls. Under the shared replay interface, the explicit selector reproduces the task-level decisions of the LLM selectors through an inspectable discrimination rule.

5.7.4 Cross-Model Agreement and the 27-Task Set

Additional completed runs with deepseek-flash extend the observed agreement beyond the original hosted model: ReAct and Reflexion complete 36/36 runs at the two-call budget, and their task-level sequences match the explicit selector. Supplementary Tables S21–S22 report the remaining model and configuration checks.

On the 27-task set, the selectors differ primarily in when they stop. ReAct completes 72/81 runs, while Reflexion resolves eight of the nine ReAct early stops and completes 80/81. The explicit rule resolves the four tasks on which ReAct stops early and uses fewer assigned calls per run (0.93 versus 1.07 for ReAct and 1.19 for Reflexion). Under the shared adjudication interface, the observed difference lies in whether a selector abstains, stops early or continues after reflection.

Figure 8 summarises the 12-task budget comparison and probe-description ablation reported in Sections 5.7.2 and 5.7.3.

6 Discussion

6.1 Implications for Auditable Tool Selection

Our planning layer links each expected tool return to the explanations and propositions it can change. The replay shows that evidence-layer ordering is decisive under tight budgets, and the shared LLM comparison shows that an explicit selector can reproduce the same action sequences through an inspectable rule. The practical contribution is an auditable selection interface: observations remain tied to provenance, candidate actions can be compared before execution and unresolved propositions remain explicit.

Table 8 gives a qualitative architectural comparison with fixed workflows, retrieval frameworks and reasoning agents. It identifies differences in state representation, action selection and feedback, and it stands apart from the measured retrieval contrasts in Tables 4 and 5.

The rows describe method specifications and the available implementation, and Section 5 reports the component results and retrospective analyses that follow from them.

6.2 How Retrieval and Content Comparison Support Planning

Cross-encoder reranking provides the primary retrieval gain. It raises Recall@5 by 11.97 percentage points over BM25 and by 20.27 points over unreranked RRF; in the lowest BM25 quartile, the gain reaches 40.51 points. Graph augmentation preserves the aggregate performance of the reranked stack while providing a targeted relation path for the small subset of queries whose missing evidence is relational.

The 6,212 identical-content pairs cover 99.90% of entries in both related builds, while byte-volume analysis localises most change to four large entries. Entry overlap identifies widespread resource continuity; byte concentration locates substantive change in bytecode, compiled resources and metadata. Code, configuration and distribution evidence connect those content-level observations to application- and operator-level interpretation.

6.3 Frozen Replay and Prospective Controlled Evaluation

A prospective evaluation can extend the same interface to newly acquired returns by freezing initial evidence, available tool interfaces, budgets and scoring before execution. Fixed workflows, conventional agents and the proposed selector would receive identical materials

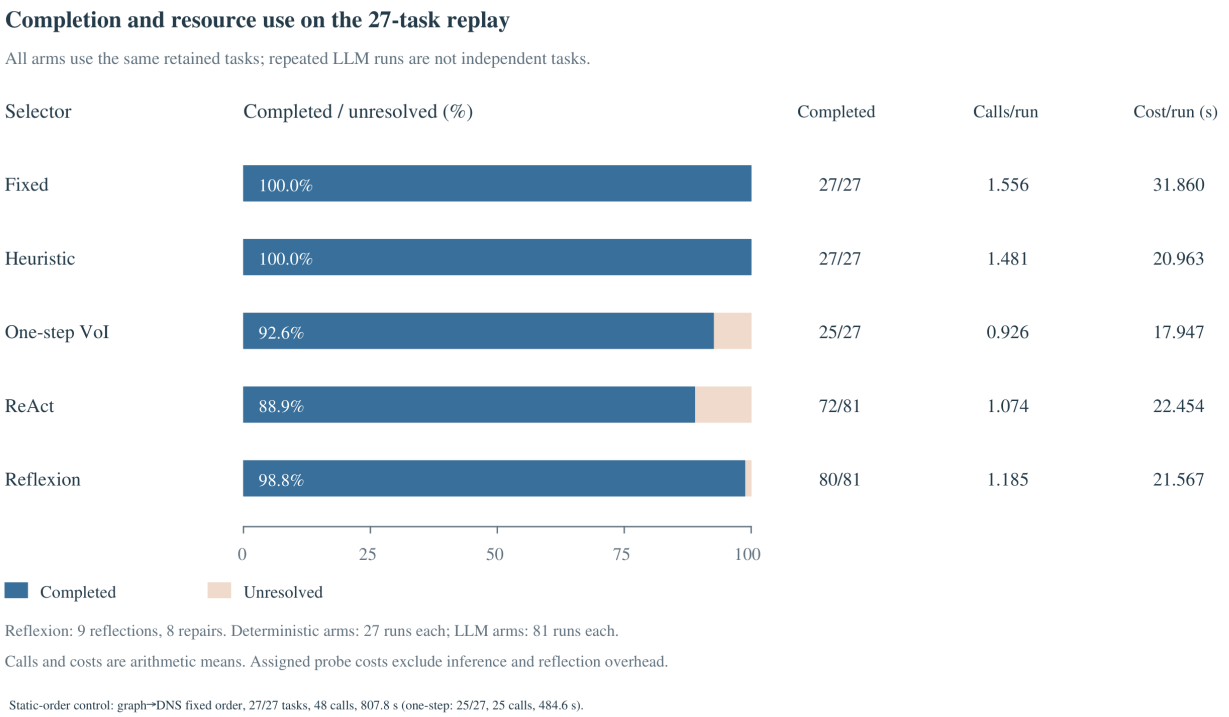


Fig. 7 Completion, call count and assigned probe cost on the 27-task sealed-return set. Deterministic arms have one run per task; ReAct and Reflexion have three repetitions per task. Stacked bars distinguish completed and unresolved runs, and labels retain the run denominators. Calls and assigned probe costs are reported as per-run means. The footer reports the static graph→DNS control for call and cost comparison

and tool responses, allowing prediction quality, action choice, cost and proposition-scoped revision to be assessed under matched conditions. The complete protocol is provided in Supplementary Appendix A.

6.4 Evaluation Scope

The evaluation establishes three layers: the framework specification, the observation pipeline and a bounded action-selection replay. Section 6.3 and Supplementary Appendix A define the matched protocol for the next prospective stage, which integrates online prediction with dependency-scoped revision.

The evidence base comprises 39 tasks across two sealed-return sets, a three-task mechanism demonstration, 87 retrieval queries, three application builds, one operational trace and one migration case. These materials support the component- and replay-level claims and provide a fixed basis for independently sampled investigations and integrated online execution.

7 Conclusions

We introduce a symbolic world-model-based framework that makes tool selection in fraud-related website investigation explicit and auditable. Across 87 queries, rank fusion with cross-encoder reranking improves Recall@5 over BM25 by 11.97 percentage points overall and by 40.51 points in the lowest BM25 quartile. Cross-build content comparison finds identical counterparts for 99.90% of entries and localises most byte-level change to four large package entries. Under a two-call budget, evidence-layer ordering completes all 12 replay tasks, while the explicit selector matches LLM task-level decisions through a transparent discrimination rule. On the 27-task set, it resolves 25 tasks in 25 calls and returns Unknown when no remaining observation can separate the scenarios. The framework's central contribution is an auditable decision interface that links evidence gaps, expected tool returns, action cost and proposition-scoped revision.

Online Resource 1: The accompanying file contains the annotation protocol and per-pair labels, agreement statistics and source digests, APK content comparisons, replay protocols and additional results (Tables S1–S23), retrieval distributions and configuration metrics (Figures S1 and S2), and three appendices: Appendix A (prospective protocol), Appendix B (sealed-return task sets, scoring conditions and implementation provenance)

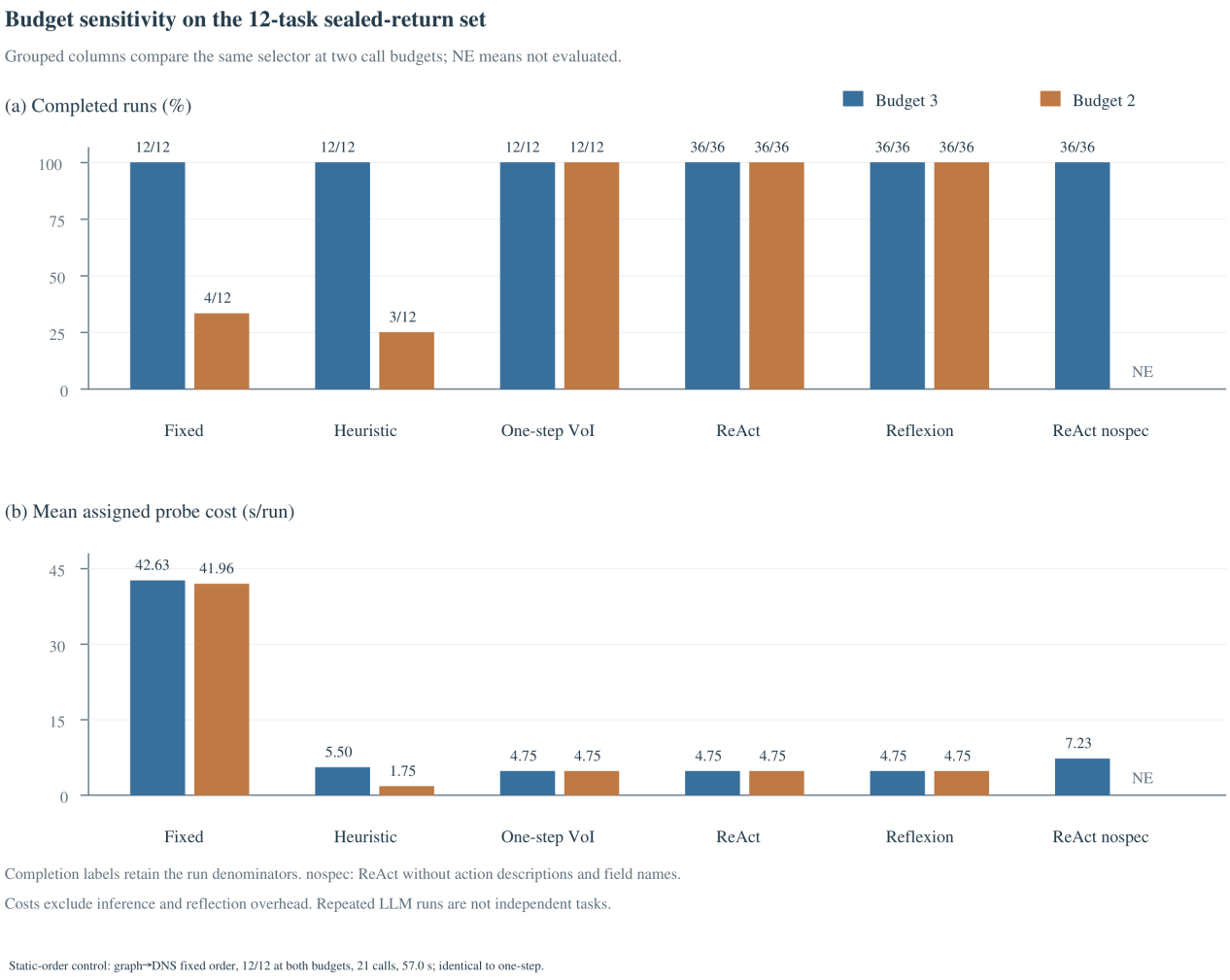


Fig. 8 Budget sensitivity on the 12-task sealed-return set. Grouped columns show completed runs and mean assigned probe cost at call budgets of three and two. Completion labels report completed runs over all runs: 12 for each deterministic arm and 36 for each LLM arm and budget. ReAct nospec is evaluated at budget three only. The footer reports the static graph→DNS control

and Appendix C (query-level and configuration-level retrieval results).

Statements and Declarations

Author contributions: J.L. and Q.F. conceived the study; J.L. designed the framework, implemented the software and carried out the experiments; Q.F. supervised the work, acquired funding and revised the manuscript. Both authors read and approved the final manuscript.

Acknowledgement: This research was supported by the 2025 Liaoning Provincial Social Science Planning Fund (grant number L25ASH004), the 2025 Basic Research Fund for Universities of the Liaoning Provincial Department of Education (grant number LJ112510175008), and the 2024 Liaoning Provincial Department of Education Project (grant number LJ112410175013). The funders had no role in the design of the study, in the collection, analysis or interpretation of data, or in the writing of the manuscript.

Ethical approval: Not applicable. The study analysed archived operational records and publicly available application samples; no human participants or animals were involved, and informed consent was therefore not required.

Data availability: The raw investigation records are restricted because they contain sensitive case material. Supplementary Tables S1–S23 provide annotation, content-comparison and replay results; Supplementary Appendices A–C provide the prospective protocol, sealed-return task and scoring specifications, implementation provenance, and query-level retrieval results. Subject to the applicable restrictions, the corresponding author can provide anonymised evaluation tables,

Table 8 Qualitative comparison of website-verification and action-planning mechanisms

Approach	State or evidence representation	Next-action mechanism	Feedback and role in this study
Fixed verification workflow	Inputs and an ordered checklist	Tool order is specified in advance	Collects outputs at prescribed steps; a reference workflow.
RAG / GraphRAG [18,20]	Retrieved documents and graph-organised relations	Retrieves and ranks material for response generation	Supplies context for generation; the retrieval components are evaluated here.
ReAct [30]	Textual reasoning and an action–observation trace	Interleaves language-model reasoning with actions	Uses returned observations in later reasoning; a reference agent architecture.
RAP [12]	Language-model states, predicted outcomes and task rewards	Searches reasoning trajectories using Monte Carlo tree search	Evaluates branches through predicted states and rewards; a reference planning method.
Existing project tool agent	Session messages, tool calls and response previews	Selects tools and arguments within the available interfaces and permissions	Adds tool returns to the message history; supplies the archived traces.
Proposed world-model-based framework	Structured B(k), provenance, conditions, propositions and dependencies	Ranks feasible actions by expected scenario-pair discrimination per unit cost	Compares expected and observed outcomes and revises affected propositions; a proposed planning design.

annotation materials, retrieval configurations and figure-generation scripts on reasonable request.

Competing interests: The authors declare no conflicts of interest.

Abbreviations

APK, Android application package; CE, cross-encoder; DNS, Domain Name System; JSON, JavaScript Object Notation; LLM, large language model; MCP, Model Context Protocol; MRR, mean reciprocal rank; RAG, retrieval-augmented generation; RRF, reciprocal rank fusion; TOST, two one-sided tests; VoI, value of information.

References

1. Alhogail, A., Alharbi, R.A. Effective ML-Based Android Malware Detection and Categorization. Electronics 2025, 14, 1486. https://doi.org/10.3390/electronics14081486
2. Alsakarnah, R., Masoud, M.Z., Ghababsheh, A. Hybridizing Explainable AI (XAI) for Intelligent Feature Extraction in Phishing Website Detection. Electronics 2026, 15, 350. https://doi.org/10.3390/electronics15020350
3. Beijing Academy of Artificial Intelligence (BAAI). bge-reranker-v2-m3: Model Card. https://huggingface.co/BAAI/bge-reranker-v2-m3. Accessed on 16 October 2026
4. Chen, W., Song, P., Lu, Q., Cao, L., Cheng, B., Feng, W., Liu, C. ReliHarness: A Self-Learning Reliability Harness for LLM Agent Tool Execution. Electronics 2026, 15, 4166. https://doi.org/10.3390/electronics15184166
5. Cormack, G.V., Clarke, C.L.A., Buettcher, S. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009, 758-759. https://doi.org/10.1145/1571941.1572114
6. Ding, J., Zhang, Y., Shang, Y., Zhang, Y., Zong, Z., Feng, J., Yuan, Y., Su, H., Li, N., Sukiennik, N., Xu, F., Li, Y. Understanding World or Predicting Future? A Comprehensive Survey of World Models. ACM Computing Surveys 2025, 58(3), 1-38. https://doi.org/10.1145/3746449
7. Doyle, J. A truth maintenance system. Artificial Intelligence 1979, 12, 231-272. https://doi.org/10.1016/0004-3702(79)90008-0
8. Dror, R., Baumer, G., Shlomov, S., Reichart, R. The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018, 1383-1392. https://doi.org/10.18653/v1/P18-1128
9. Ghioni, R., Taddeo, M., Floridi, L. Open source intelligence and AI: A systematic review of the GELSI literature. AI & Society 2024, 39, 1827-1842. https://doi.org/10.1007/s00146-023-01628-x
10. Hafner, D., Lillicrap, T., Ba, J., Norouzi, M. Dream to Control: Learning Behaviors by Latent Imagination. Proceedings of the International Conference on Learning Representations (ICLR), 2020. https://iclr.cc/virtual_2020/poster_S1lOTC4tDS.html. Accessed on 16 October 2026
11. Hafner, D., Pasukonis, J., Ba, J., Lillicrap, T. Mastering diverse control tasks through world models. Nature 2025, 640, 647-653. https://doi.org/10.1038/s41586-025-08744-2
12. Hao, S., Gu, Y., Ma, H., Hong, J., Wang, Z., Wang, D., Hu, Z. Reasoning with Language Model is Planning with World Model. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, 2023, 8154-8173. https://doi.org/10.18653/v1/2023.emnlp-main.507
13. Heuer, R.J. Psychology of Intelligence Analysis. Center for the Study of Intelligence, Washington, DC, USA, 1999. https://www.cia.gov/resources/csi/books-monographs/psychology-of-intelligence-analysis-2/. Accessed on 19 October 2026
14. Hofweber, T., Hase, P., Stengel-Eskin, E., Bansal, M. Are language models rational? The case of coherence norms and belief revision. Artificial Intelligence 2026, 361, 104639. https://doi.org/10.1016/j.artint.2026.104639

15. Howard, R.A. Information value theory. IEEE Transactions on Systems Science and Cybernetics 1966, 2, 22-26. https://doi.org/10.1109/TSSC.1966.300074
16. Jaffal, N.O., Alkhanafseh, M., Mohaisen, D. Large Language Models in Cybersecurity: A Survey of Applications, Vulnerabilities, and Defense Techniques. AI 2025, 6, 216. https://doi.org/10.3390/ai6090216
17. Kaelbling, L.P., Littman, M.L., Cassandra, A.R. Planning and acting in partially observable stochastic domains. Artificial Intelligence 1998, 101, 99-134. https://doi.org/10.1016/S0004-3702(98)00023-X
18. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-T., Rocktäschel, T., Riedel, S., Kiela, D. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems 2020, 33, 9459-9474. https://proceedings.neurips.cc/paper_files/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html. Accessed on 16 October 2026
19. Moreau, L., Missier, P. (Eds.) PROV-DM: The PROV Data Model. W3C Recommendation, 30 April 2013. https://www.w3.org/TR/2013/REC-prov-dm-20130430/. Accessed on 16 October 2026
20. Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H., Hong, C., Zhang, Y., Tang, S. Graph Retrieval-Augmented Generation: A Survey. ACM Transactions on Information Systems 2025, 44(2), 1-52. https://doi.org/10.1145/3777378
21. Qu, C., Dai, S., Wei, X., Cai, H., Wang, S., Yin, D., Xu, J., Wen, J.-R. Tool learning with large language models: a survey. Frontiers of Computer Science 2025, 19, 198343. https://doi.org/10.1007/s11704-024-40678-2
22. Robertson, S., Zaragoza, H. The Probabilistic Relevance Framework: BM25 and Beyond. Foundations and Trends in Information Retrieval 2009, 4, 1-174. https://doi.org/10.1561/1500000019
23. Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., Yao, S. Reflexion: Language Agents with Verbal Reinforcement Learning. Advances in Neural Information Processing Systems 2023, 36, 8634-8652. https://doi.org/10.52202/075280-0377
24. Thorne, J., Vlachos, A., Christodoulopoulos, C., Mittal, A. FEVER: A Large-scale Dataset for Fact Extraction and VERification. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), 2018, 809-819. https://doi.org/10.18653/v1/N18-1074
25. Tu, T., Zhang, H., Hu, Y., Cui, D., Zhai, X. A detector for Android repackaged applications with layout-fingerprint. Journal of Information Security and Applications 2023, 76, 103527. https://doi.org/10.1016/j.jisa.2023.103527
26. Varshney, G., Kumawat, R., Varadharajan, V., Tupakula, U., Gupta, C. Anti-phishing: A comprehensive perspective. Expert Systems with Applications 2024, 238, 122199. https://doi.org/10.1016/j.eswa.2023.122199
27. Xu, Y., Zhuang, Y., Liu, X., Zhang, T., Xiao, B., Xu, X., Jiang, D., Wang, J., Hu, H. LLM agents security duality: a comprehensive survey of self-security and empowered cybersecurity. Artificial Intelligence Review 2026, 59, 174. https://doi.org/10.1007/s10462-026-11563-0
28. Yadav, A., Kumar, A., Singh, V. Open-source intelligence: A comprehensive review of the current state, applications and future perspectives in cyber security. Artificial Intelligence Review 2023, 56, 12407-12438. https://doi.org/10.1007/s10462-023-10454-y
29. Yadav, T., Masum, M. Explainable Multi-Agent LLM Framework for Phishing Email Detection via Role-Specialized Evidence Decomposition. Electronics 2026, 15, 2606. https://doi.org/10.3390/electronics15122606
30. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y. ReAct: Synergizing Reasoning and Acting in Language Models. Proceedings of the International Conference on Learning Representations (ICLR), 2023. https://openreview.net/forum?id=WE_vluYUL-X. Accessed on 16 October 2026
31. Yuan, Y., Apruzzese, G., Conti, M. Beyond the west: Revealing and bridging the gap between Western and Chinese phishing website detection. Computers & Security 2025, 148, 104115. https://doi.org/10.1016/j.cose.2024.104115
32. Yumuşak, S. An Information-Theoretic Framework for Retrieval-Augmented Generation Systems. Electronics 2025, 14, 2925. https://doi.org/10.3390/electronics14152925
33. Zemplenyi, M., Miller, J.W. Bayesian Optimal Experimental Design for Inferring Causal Structure. Bayesian Analysis 2023, 18(3), 929-956. https://doi.org/10.1214/22-ba1335
34. Zhang, K., Kong, C., Wang, S., Rocha, A., Li, H. Image Provenance Analysis via Graph Encoding With Vision Transformer. IEEE Transactions on Information Forensics and Security 2025, 20, 4422-4437. https://doi.org/10.1109/tifs.2025.3559787
35. Zhang, W., Huang, J., Bi, Z., Wang, B. Flow-RAG: Retrieval-augmented generation for knowledge graph question answering via gated flow propagation. Knowledge-Based Systems 2026, 348, 116400. https://doi.org/10.1016/j.knosys.2026.116400
36. Zhao, C., Liu, G., Zhang, R., Liu, Y., Wang, J., Kang, J., Niyato, D., Li, Z., Shen, X., Han, Z., Sun, S., Yuen, C., Kim, D.I. Edge General Intelligence Through World Models, Large Language Models, and Agentic AI: Fundamentals, Solutions, and Challenges. IEEE Transactions on Cognitive Communications and Networking 2026, 12, 5649-5675. https://doi.org/10.1109/TCCN.2026.3658762