Wayfinder: Adaptive Resource Routing from Agent Citations

Miguel Romero*
University of Minnesota
romer333@umn.edu

George Karypis
University of Minnesota
karypis@umn.edu

Abstract

To act effectively, agents must gather contextual information across resources such as chats, issue trackers, email and knowledge bases, to answer questions like *what was decided in the last project meeting?* or *where is the FAQ for our internal tooling?* Where this information lives in a user's environment is unknown to the agent, so it re-explores the same resources across consecutive interactions. Common routing approaches represent each resource as a tool and rely on high-level descriptions to generalize to unseen environments. Such descriptions often fail to separate similar or overlapping resources and cannot improve from the agent's subsequent interactions. We introduce Wayfinder, an adaptive router that uses the documents cited by the downstream agent as feedback about where relevant information resides. These citations update Wayfinder's internal resource representations, which subsequent interactions use to prioritize the resources likely to contain relevant information. We also introduce resource augmentation during offline reinforcement learning, which anonymizes and permutes resource identities to discourage training-time overfitting of which resource holds which topic; our ablations suggest it helps most when few training environments are available. Because no openly available professional environments pair realistic queries with resource-access patterns, we construct and release three controlled testbeds derived from existing datasets: NFCorpus, FEVER and Tech Startup. Compared with supervised, tool-retrieval and contextual bandit baselines under a shared closed-loop protocol, Wayfinder gives the best routing when resources contain semantically overlapping content that static descriptions cannot distinguish, while description-based tool retrieval suffices when resources are drawn from distinct corpora.

1 Introduction

Large Language Models (LLMs) are increasingly judged by their ability to act as agents to accomplish tasks on behalf of users [5, 11, 30]. To complete tasks, however, an agent must do more than reason and act: it requires contextual information about the user's environment. Answering *what was decided in the last project meeting?* or *where do I look up the FAQ for internal LiteLLM?* means seeking evidence across resources such as chats, issue trackers, email, or knowledge bases, each reachable only through a tool that retrieves the documents that specific resource contains (e.g., conversations, tickets, articles) [1, 2]. Where that grounding information resides is user- or organization-specific and is not known to the agent in advance: an ongoing project may be documented in chat threads in one organization, but captured in email conversations or issue tickets in another. This creates a gap between the agent's prior knowledge and the target environment (i.e., the collection of available resources) in which it operates. Lacking that knowledge, the agent must rediscover the layout of the environment during every new user-agent interaction, re-exploring the same resources across consecutive interactions even after earlier ones revealed where some of that information is kept.

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

A natural response is to treat each resource together with its access tool as a distinct tool, which frames resource routing as tool selection problem. The distinction is that these tools while having a different resource name often have the same search functionality `[resource_name].search(query)`$\rightarrow$`docs`, and differ only in the documents behind them. Thus, any description used for selection must stand in for contents that cannot be enumerated ahead of deployment. The dominant approaches consume exactly such descriptions: an LLM ranks resources from them (ReSLLM [34]), or a contrastively trained bi-encoder embeds them and scores them by similarity to the query (ToolRetriever [26]). Both inherit the limits of the description: they cannot separate resources whose contents overlap semantically, and they are fixed at deployment, so what the agent discovers in one interaction cannot inform the next. Reinforcement learning has been applied to tool selection [25], but reward computation requires gold tool calls which are not available at deployment. Supervised routers such as Learn2Rank [12] and RouteRAG [15] share similar limitations requiring labeled queries of the target user's environment but a newly encountered environment supplies neither. What remains missing is a routing method that can adapt to a target environment without labels from it.

User–agent interactions already supply a proxy for these labels: answering a query, the agent cites the documents it relied on, and those citations are grounded evidence of where the information a user needs resides in their environment. We introduce Wayfinder, an adaptive router that updates itself from those cited documents. Wayfinder maintains a bounded representation of each resource, which we call a *resource sketch*, that subsequent interactions consult to prioritize the resources likely to contain the answer. Adaptation therefore happens online, without target-environment labels and without access to the resources' full contents. Routing is advisory rather than restrictive: the agent remains free to search beyond the recommended resources, so the signal continues to arrive even when the router is wrong. The policy itself is trained offline with Reinforcement Learning [27] on training environments; to keep it from memorizing which resource names carry which topics rather than routing on evidence, we anonymize and permute resource identities during training. Our ablations indicate this augmentation matters most when a limited number of training environments is available.

We evaluate in controlled federated environments spanning intra-corpus partitions (NFCorpus, FEVER) and cross-corpus heterogeneity (Tech Startup), with every router run under the same closed-loop protocol. Where resource descriptions read alike, citation-driven adaptation gives the best routing (Table 2): $F1_r$ of 47.67 on NFCorpus and 64.72 on FEVER against 33.03 and 31.01 for the strongest baselines, while cutting the agent's search calls by 61.2% and 73.7%. On Tech Startup, where five disjoint corpora make each resource recognizable from its description, ToolRetriever, a bi-encoder trained on training-environment labels, leads with 98.86, and Wayfinder is second with 86.60.

Our contributions are as follows:[2]

- **Problem formulation:** we formalize the *Adaptive Agent Routing Problem*, in which routing policies must be learned with each resource reachable only as a black-box retriever, weak trajectory-level feedback, and without target-environment supervision, while generalizing across environments.
- **Citation-gated supervision and method:** we introduce a supervision paradigm that updates the routing memory only from documents *cited* by the agent, filtering out the noisy retrievals that dominate early exploration (cited rate 74–95%, cited-doc accuracy 71–73% in our trajectories). We instantiate this paradigm in Wayfinder, a zero-shot federated router combining bounded resource sketches with a parameter-efficient LoRA (rank 16) policy trained via RL.
- **Generalization mechanism:** we introduce *session-randomized resource augmentation* during RL post-training. Resource identifiers are anonymized and re-permuted per session, forcing the policy to read environment memory rather than memorize name–topic associations from training environments. Our ablation (Section 7.3) shows this augmentation is most useful when training environments are scarce; when many training environments are available the policy already sees enough variability and the augmentation's benefit diminishes.

[2]The dataset is released at `https://huggingface.co/datasets/miguelrc001/wayfinder`.

- **Benchmark:** To the best of our knowledge professional user environments pairing realistic queries with resource-access patterns are not openly available, and no public evaluation captures routing with historical feedback and cross-environment generalization. We therefore construct and release three controlled testbeds derived from existing natural-language datasets to approximate such environments: *Tech Startup*, a cross-corpus setting (5 environments, 5 resources drawn from 5 distinct real corpora: emails, chat, issues, wiki, code), and *NFCorpus* [6] and *FEVER* [32], two intra-corpus settings (18 environments in which resources partition a single BEIR corpus [31]).

2 Problem Formalization

Given a sequence of user queries $\mathcal{Q} = \langle q_1, q_2, ..., q_T \rangle$ issued sequentially to an agent π, and a collection of N resources $\mathcal{R} = \{r_1, \ldots, r_N\}$ (e.g., chats, issue trackers, email, knowledge bases, etc.) we say that a resource r_j is *relevant* to query q_t iff it contains at least one document $d \in r_j$ that grounds the answer to q_t. We refer to the collection of relevant resources as $\bar{\mathcal{R}}_t \subseteq \mathcal{R}$.

Our objective is to define a routing function that, given a query q_t and the history of previous interactions between the user and the agent, identifies the resources relevant to q_t and shares them with the agent π to make exploration more efficient. We formalize this as the construction of a router f_{route} such that:

$$\hat{\mathcal{R}}_t \;=\; f_{route}(q_t,\, \mathcal{R},\, \mathcal{H}_{t-1}) \tag{1}$$

where $\mathcal{H}_{t-1} = \langle (q_i, \tau_i) \rangle_{i<t}$, is the interaction history before t, τ_i is the trajectory the agent produced at round i and $\hat{\mathcal{R}}_t \subseteq \mathcal{R}$ is the set of predicted resources. In this work, we study this problem under the following constraints:

1. **No access to target-environment supervision.** During training, the router is learned from a set of training environments and their associated queries, but the target environment and its queries are not available. This setting reflects practical deployments of agentic systems (e.g., Claude Cowork [1]) where models must generalize to previously unseen workspaces without access to user-specific data at training time.
2. **Restricted resource access via tools.** Resources are accessible only through their corresponding tools or APIs, and their contents cannot be exhaustively enumerated or randomly sampled. Consequently, methods that assume full corpus access or centralized indexing across resources are not applicable in this setting.

These constraints define a learning problem in which routing decisions must be inferred under partial observability and weak supervision, distinguishing it from standard tool use settings where the tool description and functionality does not vary across environments (bash, calculator, git, etc.). Because the router must generalize from training environments to previously unseen target deployments, we refer to this routing problem as the **Adaptive Agent Routing Problem**.

3 Related Work

End-to-end RL-based RAG systems. Recent work optimizes retrieval-augmented generation pipelines end-to-end with structured or verifiable rewards, jointly training query reformulation, retrieval, reranking, or knowledge-graph traversal [8, 16]. These systems assume direct control over the retrieval stack. Wayfinder instead isolates the upstream problem of *resource routing*, treats each resource as a black-box retriever, and acts as a parameter-efficient routing layer that can be composed with such pipelines rather than replacing them.

Resource selection and bandit routing. Resource selection has a long history in distributed information retrieval and federated search [28, 4]. Classical methods like CORI [7] use term statistics with no learning. Supervised approaches [15, 12, 17, 9] learn classifiers or rankers from labeled query–resource pairs and perform well in data-rich, stationary deployments, but assume stable resource semantics and degrade under distribution shift or in low-data target environments. A complementary line formulates routing as a bandit problem with online reward attribution [3, 14]. In agentic settings,

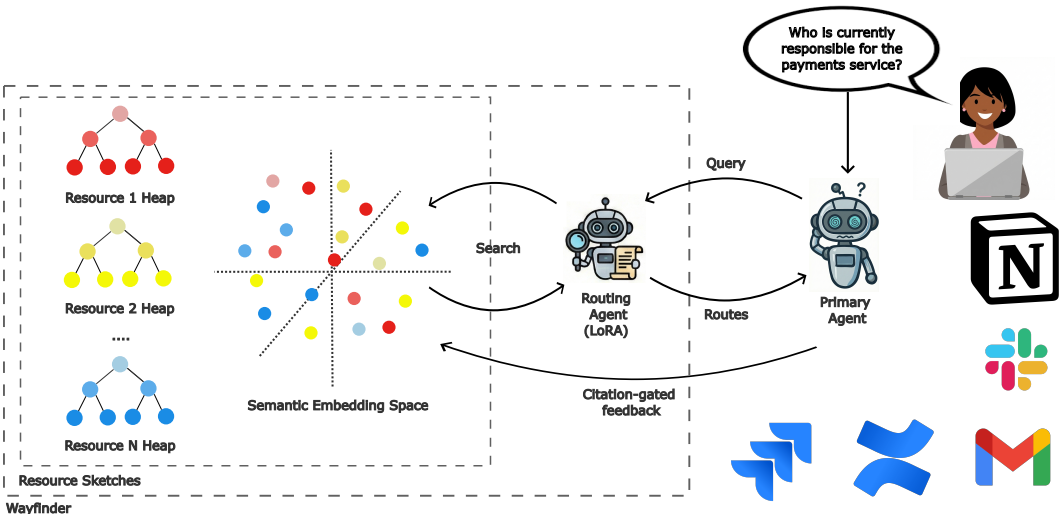


Figure 1: System overview of Wayfinder. A LoRA-adapted routing policy interacts with a bounded environment memory of *resource sketches* to select resources for downstream tool invocation, and updates its routing behavior based on citation-derived feedback from the primary agent.

however, feedback is sparse, trajectory-level, and mediated by multi-step tool use, complicating per-arm credit assignment, and standard bandit formulations maintain only compact per-arm statistics rather than structured interaction context.

Tool retrieval and zero-shot tool generalization. Treating each resource together with its access tool as a distinct tool makes our action space formally an instance of tool selection with the key distinction that descriptions, when used, have to reflect the content of the resources as the functionality and signature are largely the same across tools (to retrieve documents and `search(query)`$\rightarrow$`docs`). ReSLLM [34] is a method that uses a description of the resource contents to describe the tool which an LLM (used as the router) uses to rank the tools. On the other hand dense-retrieval variants like ToolRetriever [26] embeds the same descriptions with a contrastively trained bi-encoder and scores them by similarity to the query .

Reinforcement learning has also been applied to tool selection [25], with the reward computed by matching the predicted call against a gold tool call. Learning therefore requires labeled data for these tools. A newly encountered target environment, with a new distribution of resources, carries no such annotation: no query is labelled with the resource that holds its answer. Wayfinder instead rewards a routing decision using the documents the primary agent cites while answering requiring no labelled data of the target environment.

Memory-augmented agents. Long-term memory systems for LLMs (MemoryLLM [35], M+ [36], MemGPT [24]) and agentic continual-learning systems such as CLIN [23] compress past interactions to retain content or user preferences. These approaches target *content retention* rather than the *navigational* problem of identifying which external resources are likely to be useful for a query.

What separates Wayfinder from the work above is its supervision, not its components. Most routers discussed decide from evidence fixed before deployment: a resource description (ReSLLM, ToolRetriever), corpus statistics over enumerated contents (CORI, ReDDE), or labelled query–resource pairs from the target environment (Learn2Rank, RouteRAG). Wayfinder instead uses the agent's citations to adapt to the target environment and become more efficient as the user interacts more with the agent.

4 Methodology: Wayfinder

When we open a new agent session and ask it to perform an action or answer a question the agent often needs to explore the environment to gather context, learning along the way where the information that

is relevant to that user is kept. This knowledge is lost when the session ends, and the next session has to repeat the same exploration. We argue that this is inefficient. Queries about the same project tend to be answered from the same few resources, so earlier sessions can inform later ones. Citations also point at evidence the agent actually used to answer a question. Wayfinder builds upon this observation and proposes a citation-gating mechanism to retain and utilize this knowledge across sessions.

Wayfinder is a routing method that is paired with an agent π to make its trajectories more efficient over time. When a new query is passed to π it can call Wayfinder to retrieve the resource where the information is likely to be found. On the other hand, when a trajectory from the agent π is completed, this is passed to Wayfinder to store the knowledge. At a high level Wayfinder is composed of an environment memory $\mathcal{M}$, containing an internal representation of the environment, and a LLM-based routing policy π_{route} that uses that memory to provide resource recommendations to the agent π. Algorithm 1 states the loop in full: routing (line 3), the agent's rollout (line 5), citation extraction (lines 8–10) and the sketch update (lines 12–19).

4.1 Citation gating

A document is written to the memory $\mathcal{M}$ only if the agent *cited* it and its identifier appears in one of the tool outputs. Formally, given the cited document identifiers I_t and the tool-call log $\{(r_{i,k}, D_{i,k})\}_k$ of the agent's trajectory τ_t, where the k-th pair records the resource searched $r_{i,k}$ and the documents it returned $D_{i,k}$, we define the verified citation set $\mathcal{C}_t$ as the documents that are both cited and retrieved:

$$\mathcal{C}_t \;=\; \Big\{\, d \;:\; \mathit{id}(d) \in I_t \;\wedge\; d \in \textstyle\bigcup_k D_{i,k} \,\Big\}. \tag{2}$$

Identifiers matching no retrieved document are discarded. We write $\rho(\mathcal{C}_t)$ to note the set of resources these documents came from:

$$\rho(\mathcal{C}_t) \;=\; \Big\{\, r_{i,k} \;:\; \exists\, d \in \mathcal{C}_t \cap D_{i,k} \,\Big\}. \tag{3}$$

If the agent terminates with no cited documents, $I_t = \emptyset$, then $\mathcal{C}_t = \emptyset$ and no update occurs. Restricting $\mathcal{M}$ to documents that were cited and appear in a tool output keeps evidence the agent judged relevant to the user, rather than material it merely explored, and filters out hallucinated documents.

We characterize this signal over a query stream with two statistics: the *citation rate* $\Pr_t[\mathcal{C}_t \neq \emptyset]$, how often citations are available, and the *citation accuracy* $\mathbb{E}_t[\,|\rho(\mathcal{C}_t) \cap \bar{\mathcal{R}}_t|/|\rho(\mathcal{C}_t)| \mid \mathcal{C}_t \neq \emptyset\,]$, how often they name the right resource given that a citation exists, where $\bar{\mathcal{R}}_t$ is the ground-truth resource set for q_t.

4.2 Resource Sketches

Evidence is stored per resource. For each $r_j \in \mathcal{R}$ the *resource sketch* $\mathcal{M}^j$ is a bounded set of at most M records, each a triple (d, q, t): a cited document d observed via a tool call to r_j, the query q that elicited the citation, and an access timestamp. The *environment memory* is then the collection of these resource sketches $\mathcal{M} = (\mathcal{M}^1, \ldots, \mathcal{M}^N)$. Sketches start empty, $\mathcal{M}^j_0 = \emptyset$, so on the first query the policy may emit an empty action set and let the agent fall back on static descriptions. Thus, when Wayfinder is applied to a new environment it starts with an empty memory but a trained routing policy π_{route}.

On each rollout, every document in $\mathcal{C}_t$ retrieved from a cited resource $r_j \in \rho(\mathcal{C}_t)$ is inserted into $\mathcal{M}^j$ as a record (d, q_t, t), evicting the least-recently-used record when the sketch is at capacity M. Resources outside $\rho(\mathcal{C}_t)$ are untouched. Three consequences follow: retrieved-but-uncited documents never enter, so early exploration does not pollute sketches with low-precision associations; when the agent fails to cite, sketches go stale rather than wrong, giving slower but stable adaptation; and the capacity M caps the storage and retrieval cost of the routing layer regardless of stream length. Lastly, a centralized semantic index over $\mathcal{M}$ supports retrieval across resource sketches of the top-k documents relevant to a query.

Algorithm 1: Wayfinder routing loop and citation-gated memory update

Input: Resources $\mathcal{R}$; query stream $\{q_t\}_{t=1}^{T}$; routing policy π_{route}; primary agent π; budget B; per-resource sketch capacity M.

Output: Routing decisions $\{\hat{\mathcal{R}}_t\}_{t=1}^{T}$ and final environment memory $\mathcal{M}_T$.

Initialize $\mathcal{M}_0 \leftarrow (\emptyset, \ldots, \emptyset)$;
for $t = 1, \ldots, T$ **do**
  `// (1) Routing`
  $\hat{\mathcal{R}}_t \leftarrow \textsc{Route}(\pi_{route}, q_t, \mathcal{M}_{t-1}, \mathcal{R})$; `//` $|\hat{\mathcal{R}}_t| \le B$
  `// (2) Primary-agent rollout and parsing`
  $\tau_t \leftarrow \pi(q_t, \hat{\mathcal{R}}_t)$;
  $\big(y_t, \{(r_{i,k}, D_{i,k})\}_k\big) \leftarrow \textsc{ParseTrajectory}(\tau_t)$;
  `// (3) Citation extraction (Eqs. 2-3)`
  $I_t \leftarrow$ identifiers of the documents cited in τ_t;
  $\mathcal{C}_t \leftarrow \{d : id(d) \in I_t \wedge d \in \bigcup_k D_{i,k}\}$;
  $\rho(\mathcal{C}_t) \leftarrow \{r_{i,k} : \exists\, d \in \mathcal{C}_t \cap D_{i,k}\}$;
  `// (4) Sketch update (citation-gated, LRU-bounded)`
  $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1}$;
  if $\mathcal{C}_t \neq \emptyset$ **then**
    foreach $r_j \in \rho(\mathcal{C}_t)$ **do**
      foreach $d \in \mathcal{C}_t$ *retrieved from* r_j **do**
        if $|\mathcal{M}_t^j| \ge M$ **then**
          Evict LRU record from $\mathcal{M}_t^j$;
        $\mathcal{M}_t^j \leftarrow \mathcal{M}_t^j \cup \{(d, q_t, t)\}$;
    Update centralized semantic index over $\mathcal{M}_t$;

4.3 Routing

Given q_t, the routing policy π_{route} reads the environment memory state at time $t-1$, $\mathcal{M}_{t-1}$, through the semantic index in a ReAct-style loop, optionally issuing further memory searches with refined queries and reasoning over what it retrieves before emitting $\hat{\mathcal{R}}_t$. The agent then executes $\tau_t = \pi(q_t, \hat{\mathcal{R}}_t)$ and the trajectory returns for citation extraction, closing the loop.

The routing policy is implemented as a LLM trained with reinforcement learning with verifiable rewards [21] and GRPO [27] on training environments only. Specifically, for each query we sample a group of G rollouts and score rollout i by $R_i = F_1(\hat{\mathcal{R}}_t^{(i)}, \bar{\mathcal{R}}_t)$ against the ground-truth set $\bar{\mathcal{R}}_t = \{r_j \in \mathcal{R} : r_j \cap D_t \neq \emptyset\}$, where D_t is the set of relevance-judged documents for q_t; malformed output (invalid identifiers or non-parseable JSON) scores zero. The G scores are then centred and scaled against each other, $A_i = (R_i - \mu_q)/(\sigma_q + \epsilon)$, where μ_q and σ_q are the mean and standard deviation of $\{R_i\}_{i=1}^{G}$ and $\epsilon > 0$ guards the case where all G rollouts score alike; a rollout that beats its siblings on the same query receives positive weight, one that trails them negative. The policy is then updated by advantage-weighted cross-entropy [37] over the router's own generated tokens.

Labeled data D_t is only available on training environments; at testing time Wayfinder is applied to a new environment with unseen resources and adapts from citations alone.

4.4 Forcing the policy to read evidence, not identifiers

A policy trained on a small number of environments can memorize which identifier carries which topic ("coding questions go to resource A"). The shortcut scores well in training and collapses in a target environment for which resources may carry different content, so it has to be prevented during training rather than detected afterwards. We therefore apply two augmentations while training the policy, both at the granularity of a *session*: a single training rollout, within which one assignment of identifiers to resources is held fixed.

Table 1: Benchmark statistics.

Dataset	Type	# Env.	# Res. /Env.	Train / Test Queries	Doc. / Res.
NFCorpus	Intra-corpus	2	50	89 / 30	121
Fever	Intra-corpus	16	50	333.7 / 82.9	6831.8
Tech Startup	Cross-corpus	5	5	180.4 / 45.0	150.5

Intra-corpus environments partition a single BEIR corpus into semantically coherent resources via GMM clustering. *Cross-corpus* environments draw each resource from a distinct real corpus, yielding heterogeneity at the resource level rather than within a shared corpus. Doc./Res. is the average number of documents per resource; Train/Test follows an $80/20$ temporal split within each environment.

Resource encoding. Descriptive names ("email", "chat") are replaced by anonymous identifiers `resource_1`, . . . , `resource_N` in both the input and the action vocabulary, removing name-based priors.

Resource randomization. Each rollout draws a fresh permutation mapping resources to identifiers, applied consistently within that rollout.

Together the two leave no stable association between an identifier and a topic across query rollouts. Sketch content is then the only signal that survives from one rollout to the next, which is the signal that the policy must learn to interpret.

5 Dataset

Routing in real deployments combines two sources of heterogeneity: *intra-corpus partitioning*, where different organizations split similar content across different resources (e.g., release notes in the wiki vs. chat archives); and *cross-corpus heterogeneity*, where different resource types serve different communicative functions and therefore expose different vocabulary, length, and structural style (e.g., syntactic code blocks vs. conversational messages). Existing federated-search benchmarks [33, 10] contain too few environments to support train–deploy splits, and BEIR [31] provides single-corpus relevance judgments without independent environments. We therefore evaluate Wayfinder on two complementary regimes (Table 1): *intra-corpus* environments derived from BEIR, and a new *cross-corpus Tech Startup* benchmark in which each resource is drawn from a distinct real corpus.

We derive 18 environments by partitioning two BEIR corpora into disjoint resources via sentence-transformer embeddings and a Gaussian Mixture Model: the specialized NFCorpus [6] (2 environments) and the larger fact-verification FEVER corpus [32] (16 environments). Each environment inherits human-annotated document-level relevance judgments from the source dataset; queries are arranged in an $80/20$ temporal split.

We additionally construct a 5-environment *Tech Startup* benchmark in which the five resources (emails, chat channels, issue tracker, wiki pages, code repository) are sourced respectively from Enron emails [20], the Alpaca instruction-tuning corpus [29], GitHub issues from SWE-bench [19], Wikipedia paragraphs from HotpotQA [38], and source code from CodeSearchNet [18]. Documents and queries are disjoint across environments. Queries are rewritten to reference content rather than the originating application and every retained query passes a semantic-retrieval quality gate (top-5 dense retrieval). Unlike the intra-corpus environments, resource boundaries here reflect natural cross-application heterogeneity rather than synthetic clustering. Full construction and validation details are in Appendix C.

6 Experimental Setup

We report three metrics, each a percentage for which higher is better. The router's $F1_{route}$ [28, 4] which scores the accuracy of the recommendation on its own. The agent's citation $F1_{cite}$, which measures the effect of router on the agent's grounding. We do not report answer accuracy: the primary agent can answer many queries in these public benchmarks from parametric knowledge, so a correct answer does not show that relevant evidence was retrieved, whereas $F1_{cite}$ credits only answers

grounded in the judged documents. The exploration savings S which measures that effect of the router on the agent's search effort: writing $s(q)$ and $s_0(q)$ for the number of search calls the agent issues on query q with and without a router, then $S = 100 \times \text{mean}_q(1 - s(q)/s_0(q))$ is the saving on a typical query.

We compare Wayfinder against baselines spanning classical, supervised, tool-retrieval and bandit families. *CORI* [7] ranks resources from term statistics without learning. *ResLLM* [34] prompts an LLM with document-derived resource summaries. *ToolRetriever* [26] embeds the same summaries with a bi-encoder contrastively trained on training environments and scores them against the query. *LinUCB-D* [22] treats each resource as a bandit arm updated from citation-derived binary feedback. *Learn2Rank* [12] trains a supervised classifier using labels from the target environment which violates our problem assumptions but we add for reference. LinUCB-D and Wayfinder adapt during the stream; CORI, ResLLM and ToolRetriever are fixed once the environment is entered. All embedding-based methods share the same MiniLM-L6-v2 encoder.

Wayfinder routing policy is implemented by finetuning Qwen3-8B using LoRA (rank 16, $\sim$0.1% of base parameters) and resource randomization. The routing agent retrieves the top 25 memory entries per decision step via semantic search to bound its context. Unless otherwise stated, the per-resource sketch capacity is $M = 100$. Training runs 125 GRPO iterations; each iteration samples 8 queries and generates 8 rollouts per query (GRPO group size for advantage normalization).

Reported results are average k-fold performance where datasets are partitioned at the environment level ($k = 2$ for NFCorpus, $k = 4$ for FEVER and Tech Startup). Methods like ToolRetriever and Wayfinder requiring cross-environment training are trained on the training split. If the method relies on interaction statistics (e.g., CORI or Wayfinder) those are produced as test environments are evaluated. Wayfinder is evaluated closed-loop: its memory is seeded with the documents cited in the target environment's training-split trajectories, without relevance labels, and is then updated from the agent's citations as test queries are processed. As a primary agent we use GPT-4o-mini.

7 Results

In this section we present experimental results and ablations under stationary conditions. Due to space constraints the evaluations in non-stationary settings and case study are included in Appendix A and Appendix B, respectively.

7.1 Experiment 1: Routers Comparison

Our hypothesis is that citation-derived adaptation pays in proportion to how poorly static descriptions separate resources. On the intra-corpus benchmarks, roughly fifty resources are drawn from a single corpus and their summaries read alike, so a router fixed at deployment has little left to discriminate on; on Tech Startup each resource is a distinct corpus and its description is close to sufficient on its own. Every router therefore enters the target environment cold and sees the identical query stream: the adaptive routers revise their beliefs as that stream is processed while the static ones remain fixed, and each router's recommendations are logged as they are issued and scored afterwards against the same relevance judgments. Wayfinder is evaluated closed-loop: its memory is updated with the citations the primary agent emits after acting on Wayfinder's own recommendations, on the same query stream every baseline receives.

Routing budgets are B=5 on NFCorpus and FEVER and B=1 on Tech Startup. Wayfinder uses the same LoRA policy with no retraining and one adapter per fold, and FEVER is capped at 50 queries per test environment. Each replay is paired with a no-router control on the same queries, which issues 25.90, 32.90 and 5.86 search calls per query on the three datasets. $F1_c$ and Saved % are measured by hard-routed replay because the closed loop itself is *advisory*: the agent may search outside the recommendation, so its raw search counts exceed an unrouted agent's and the savings metric is negative by construction, which measures the protocol rather than the router. NFCorpus rests on n=42: its two test splits yield only 9 and 33 GT-usable queries. $F1_{route}$ uses the same RDR-based definition throughout the table; plain set recall would read 28.16 on NFCorpus, where 43% of queries emit fewer resources than there are gold ones.

On the intra-corpus benchmarks, where roughly fifty resources are drawn from one corpus, Wayfinder gives the best routing: $F1_r$ of 47.67 on NFCorpus and 64.72 on FEVER, against 33.03 (ResLLM)

Table 2: Closed-loop routing comparison.

	NFCorpus (2-fold)			Fever (4-fold)			Tech Startup (4-fold)		
Router	$F1_r \uparrow$	$F1_c \uparrow$	Saved %	$F1_r \uparrow$	$F1_c \uparrow$	Saved %	$F1_r \uparrow$	$F1_c \uparrow$	Saved %
No routing	–	14.42	0.0	–	33.02	0.0	–	87.44	0.0
CORI	29.32	**20.27**	58.4	7.90	15.83	40.6	66.22	57.78	55.3
ResLLM	<u>33.03</u>	11.51	63.5	11.77	20.90	45.8	30.16	24.18	22.8
ToolRetriever	22.71	13.91	50.4	30.38	**55.52**	37.9	**98.86**	**85.31**	70.4
LinUCB-D	26.47	15.43	49.5	9.57	18.33	−4.7	71.78	62.81	58.7
Learn2Rank	12.78	5.64	41.1	<u>31.01</u>	41.64	9.9	83.28	72.16	59.8
Wayfinder	**47.67**	<u>17.10</u>	61.2	**64.72**	<u>50.88</u>	73.7	<u>86.60</u>	<u>80.78</u>	60.4

$F1_r$: routing F1. $F1_c$: the agent's citation F1 against the relevance judgments. Saved %: reduction in the agent's search calls relative to no routing. Bold: best; underline: second best.

and 31.01 (Learn2Rank) for the strongest baselines. FEVER is the dataset where routing is closest to degenerate, since a constant router that ignores the query and selects the five largest resources scores 33.6 there. Better routing carries through to the agent: Wayfinder cuts search calls on a typical query by 61.2% and 73.7% while raising $F1_c$ over the unrestricted agent by 2.7 and 17.9 points. It does not ground the most answers, however: CORI on NFCorpus (20.27) and ToolRetriever on FEVER (55.52) reach higher $F1_c$. Saved % must be read together with $F1_c$, because a router that sends the agent nowhere useful makes it give up early and so *appears* to save exploration: ResLLM saves 63.5% on NFCorpus while its $F1_c$ falls below the unrestricted agent's (11.51 vs. 14.42).

On Tech Startup, where each resource is a distinct corpus and its description is close to sufficient on its own, ToolRetriever is best on all three metrics (98.86, 85.31 and 70.4%). Wayfinder is second on routing and citation F1 (86.60 and 80.78), ahead of Learn2Rank (83.28), which is trained on target-environment labels, and of LinUCB-D (71.78), the other router that adapts online. No router improves on letting the agent search all five resources ($F1_c$ of 87.44). LinUCB-D trails Wayfinder on routing on all three datasets and on FEVER increases the agent's search calls (−4.7% saved).

These results match the two regimes of our hypothesis: citation-derived adaptation gives the best routing where resource descriptions read alike, while description-based tool retrieval suffices where resources are distinct corpora. Although Wayfinder employs a reasoning-based routing policy with environment memory, the routing behavior observed in our experiments is consistently shallow. Across all three datasets, trajectories exhibit a median of two routing decisions per query, with over 99% of trajectories completing within two steps (one retrieval step and one action step) and no evidence of long-horizon routing.

7.2 Ablation 1: RL post-training

We isolate the contribution of RL-based fine-tuning by comparing Wayfinder with and without RL post-training while holding memory structure and resource representations constant.

Table 3: Ablation on RL post-training

	NFCorpus			FEVER		
Router	$F1_r \uparrow$	$F1_c \uparrow$	Saved %	$F1_r \uparrow$	$F1_c \uparrow$	Saved %
w/o RL	41.28	17.77	63.9	66.06	75.93	81.8
w/ RL	**50.69**	20.05	64.8	**67.94**	59.70	79.3

$F1_r$: routing F1. $F1_c$: the agent's citation F1 against the relevance judgments. Saved %: reduction in the agent's search calls relative to no routing. Bold: higher $F1_r$.

Table 3 shows RL-based post-training is critical: removing it consistently degrades routing $F1_r$ across datasets despite identical memory and resource representations, indicating that static inference over environment memory is insufficient and explicit credit assignment is required to optimize routing decisions for downstream retrieval utility. On FEVER, however, the variant without RL grounds

more answers than the full model (75.93 vs. 59.70) while also searching less, so RL post-training buys routing accuracy rather than downstream grounding there.

7.3 Ablation 2: Resource Augmentation

We evaluate the impact of resource augmentation (Section 4.4) on cross-environment transfer.

Table 4: Ablation on resource augmentation.

	NFCorpus			FEVER		
Router	$F1_r \uparrow$	$F1_c \uparrow$	Saved %	$F1_r \uparrow$	$F1_c \uparrow$	Saved %
w/o res. aug.	45.35	17.54	56.9	**68.40**	58.37	76.8
w/ res. aug.	**50.69**	20.05	64.8	67.94	59.70	79.3

$F1_r$: routing F1 at budget B=5 (Appendix G). $F1_c$: the agent's citation F1 against the relevance judgments. Saved %: reduction in the agent's search calls relative to no routing. Bold: higher $F1_r$.

Table 4 shows resource augmentation improves routing robustness on NFCorpus and slightly degrades performance on FEVER, supporting our hypothesis that augmentation is most beneficial when training environments are scarce and added variability helps prevent overfitting.

8 Limitations and future work

Wayfinder depends on the primary agent's citations. Missing citations slow adaptation, but incorrect ones write wrong evidence into the resource sketches, and routing degrades as their rate grows (Appendix F.2); weighting updates by citation confidence, decaying unconfirmed evidence, or revising a sketch when later citations contradict it could limit this, and we leave their evaluation to future work. Our benchmarks are controlled testbeds built from public corpora, so they do not capture enterprise permission structures or production-scale latency. Evaluating Wayfinder with authentic tools and users is the natural next step.

9 Conclusion

We introduced Wayfinder, a router that adapts to a target environment from the citations its primary agent emits, without target-environment labels. Under a closed-loop protocol shared by all routers, Wayfinder gives the best routing on the intra-corpus benchmarks (47.67 and 64.72 $F1_r$ against 33.03 and 31.01 for the strongest baselines) and cuts the agent's search calls by 61.2% and 73.7% while improving its citation F1 over an unrouted agent. Where resources are distinct corpora, as in Tech Startup, description-based tool retrieval suffices and ToolRetriever leads, with Wayfinder second. Closed-loop routing stays within 3.3 points of the offline protocol, and RL post-training improves routing F1 on both intra-corpus benchmarks.

References

[1] Anthropic. Cowork: Claude Code power for knowledge work, 2026. URL `https://claude.com/product/cowork`. Accessed: 2026-04-27.

[2] Anthropic. Your favorite work tools are now interactive connectors inside Claude, January 2026. URL `https://claude.com/blog/interactive-tools-in-claude`. Accessed: 2026-01-28.

[3] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002. doi: 10.1023/A:1013689704352. URL `https://doi.org/10.1023/A:1013689704352`.

[4] Krisztian Balog, Robert Neumayer, and Kjetil Nørvåg. Collection ranking and selection for federated entity search. In *International Symposium on String Processing and Information Retrieval*, pages 73–85. Springer, 2012.

[5] Chaithanya Bandi, Ben Hertzberg, Geobio Boo, Tejas Polakam, Jeff Da, Sami Hassaan, Manasi Sharma, Andrew Park, Ernesto Hernandez, Dan Rambado, Ivan Salazar, Rafael Cruz, Chetan Rane, Ben Levin, Brad Kenstler, and Bing Liu. MCP-Atlas: A large-scale benchmark for tool-use competency with real MCP servers. *arXiv preprint arXiv:2602.00933*, 2025.

[6] Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. A full-text learning to rank dataset for medical information retrieval. 2016. URL `http://www.cl.uni-heidelberg.de/~riezler/publications/papers/ECIR2016.pdf`.

[7] Jamie Callan. Distributed information retrieval. 2002. URL `https://api.semanticscholar.org/CorpusID:7940632`.

[8] Yiqun Chen, Lingyong Yan, Weiwei Sun, Xinyu Ma, Yi Zhang, Shuaiqiang Wang, Dawei Yin, Yiming Yang, and Jiaxin Mao. Improving retrieval-augmented generation through multi-agent reinforcement learning. *arXiv preprint arXiv:2501.15228*, 2025.

[9] Zhuyun Dai, Yubin Kim, and Jamie Callan. Learning to rank resources. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 837–840, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350228. doi: 10.1145/3077136.3080657. URL `https://doi.org/10.1145/3077136.3080657`.

[10] Thomas Demeester, Dolf Trieschnigg, Dong Nguyen, Djoerd Hiemstra, and Ke Zhou. Fedweb greatest hits: Presenting the new test collection for federated web search. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, page 27–28, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450334730. doi: 10.1145/2740908.2742755. URL `https://doi.org/10.1145/2740908.2742755`.

[11] Xiang Deng, Jeff Da, Edwin Pan, Yannis Yiming He, Charles Ide, Kanak Garg, Niklas Lauffer, Andrew Park, Nitin Pasari, Chetan Rane, et al. Swe-bench pro: Can ai agents solve long-horizon software engineering tasks? *arXiv preprint arXiv:2509.16941*, 2025.

[12] Ulugbek Ergashev, Eduard Dragut, and Weiyi Meng. Learning to rank resources with gnn. In *Proceedings of the ACM Web Conference 2023*, pages 3247–3256, 2023.

[13] James C. French and Allison L. Powell. Metrics for evaluating database selection techniques. *World Wide Web*, 3(3):153–163, 2000. doi: 10.1023/A:1019241915635.

[14] Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for non-stationary bandit problems. *arXiv preprint arXiv:0805.3415*, 2008.

[15] Yucan Guo, Miao Su, Saiping Guan, Zihao Sun, Xiaolong Jin, Jiafeng Guo, and Xueqi Cheng. Routerag: Efficient retrieval-augmented generation from text and graph via reinforcement learning. *arXiv preprint arXiv:2512.09487*, 2025.

[16] Chuzhan Hao, Wenfeng Feng, Yuewei Zhang, and Hao Wang. Dynasearcher: Dynamic knowledge graph augmented search agent via multi-reward reinforcement learning. *arXiv preprint arXiv:2507.17365*, 2025.

[17] Dzung Hong, Luo Si, Paul Bracke, Michael Witt, and Tim Juchcinski. A joint probabilistic classification model for resource selection. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, page 98–105, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450301534. doi: 10.1145/1835449.1835468. URL `https://doi.org/10.1145/1835449.1835468`.

[18] Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. CodeSearchNet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*, 2019.

[19] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024.

[20] Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning (ECML)*, pages 217–226. Springer, 2004.

[21] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

[22] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.

[23] Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. Clin: A continually learning language agent for rapid task adaptation and generalization. *arXiv preprint arXiv:2310.10134*, 2023.

[24] Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

[25] Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. Toolrl: Reward is all tool learning needs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[26] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations (ICLR)*, 2024.

[27] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[28] Milad Shokouhi and Luo Si. Federated search. *Foundations and Trends in Information Retrieval*, 5(1):1–102, 2011. doi: 10.1561/1500000010.

[29] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following LLaMA model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

[30] The Terminal-Bench Team. Terminal-bench: A benchmark for ai agents in terminal environments, Apr 2025. URL `https://github.com/harbor-framework/terminal-bench`.

[31] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL `https://openreview.net/forum?id=wCu6T5xFjeJ`.

[32] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *NAACL-HLT*, 2018.

[33] Shuai Wang, Ekaterina Khramtsova, Shengyao Zhuang, and Guido Zuccon. Feb4rag: Evaluating federated search in the context of retrieval augmented generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 763–773, 2024.

[34] Shuai Wang, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. Resllm: Large language models are strong resource selectors for federated search. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 1360–1364, 2025.

[35] Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, Jingbo Shang, and Julian McAuley. Memoryllm: Towards self-updatable large language models. *arXiv preprint arXiv:2402.04624*, 2024.

[36] Yu Wang, Dmitry Krotov, Yuanzhe Hu, Yifan Gao, Wangchunshu Zhou, Julian McAuley, Dan Gutfreund, Rogerio Feris, and Zexue He. M+: Extending memoryllm with scalable long-term memory. *arXiv preprint arXiv:2502.00592*, 2025.

[37] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[38] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380, 2018.

A Robustness to Non-Stationarity

Non-stationarity is a defining characteristic of federated environments, where the information within resources can change over time independently of the agent's actions. To evaluate robustness to such exogenous changes, we progressively relocate an increasing fraction of ground-truth documents associated with test queries to different resources before testing, simulating changes in where information resides after deployment. Higher drift levels subsume lower ones to ensure consistency across settings, and all methods are evaluated under identical drift realizations to isolate differences in routing behavior. We report routing recall and precision to assess how effectively each method maintains routing quality as the environment diverges from the training distribution.

Figure 2 shows that routing performance degrades for all methods as the degree of ground-truth movement increases, reflecting the difficulty of routing under severe distribution shift. However, Wayfinder maintains higher recall and precision across all drift levels, indicating greater robustness to changes in where relevant information resides. In contrast, baselines that rely on static assumptions or fixed ranking signals exhibit steeper performance declines as drift increases. These results suggest that learning routing policies from interaction feedback yields representations that are less tightly coupled to specific resource assignments, improving resilience to post-deployment environmental changes.

B Appendix: Case Study

We present a qualitative case study illustrating how Wayfinder performs resource routing using incomplete environment memory, a common setting in federated environments where memory provides only sparse and noisy evidence about resource contents.

Query.

> As of 2014, the electric chair is an optional form of execution in Alabama, Florida, South Carolina, and Idaho.

Given the query, the agent retrieves semantically related memory samples from multiple resources. Each resource contains a small subset of documents that only partially reflect its underlying collection.

```
Resource 34:
  - Hill v. McDonough (U.S. Supreme Court case on
    execution methods in Florida)
  - Capital punishment (overview of death penalty
    practices)
  - Capital punishment debate in the United States
  - List of executions by lethal injection

Resource 51:
  - Shapiro v. McManus (U.S. Supreme Court case on
    judicial procedure)
  - Jordan Sekulow (U.S. legal commentator)
```

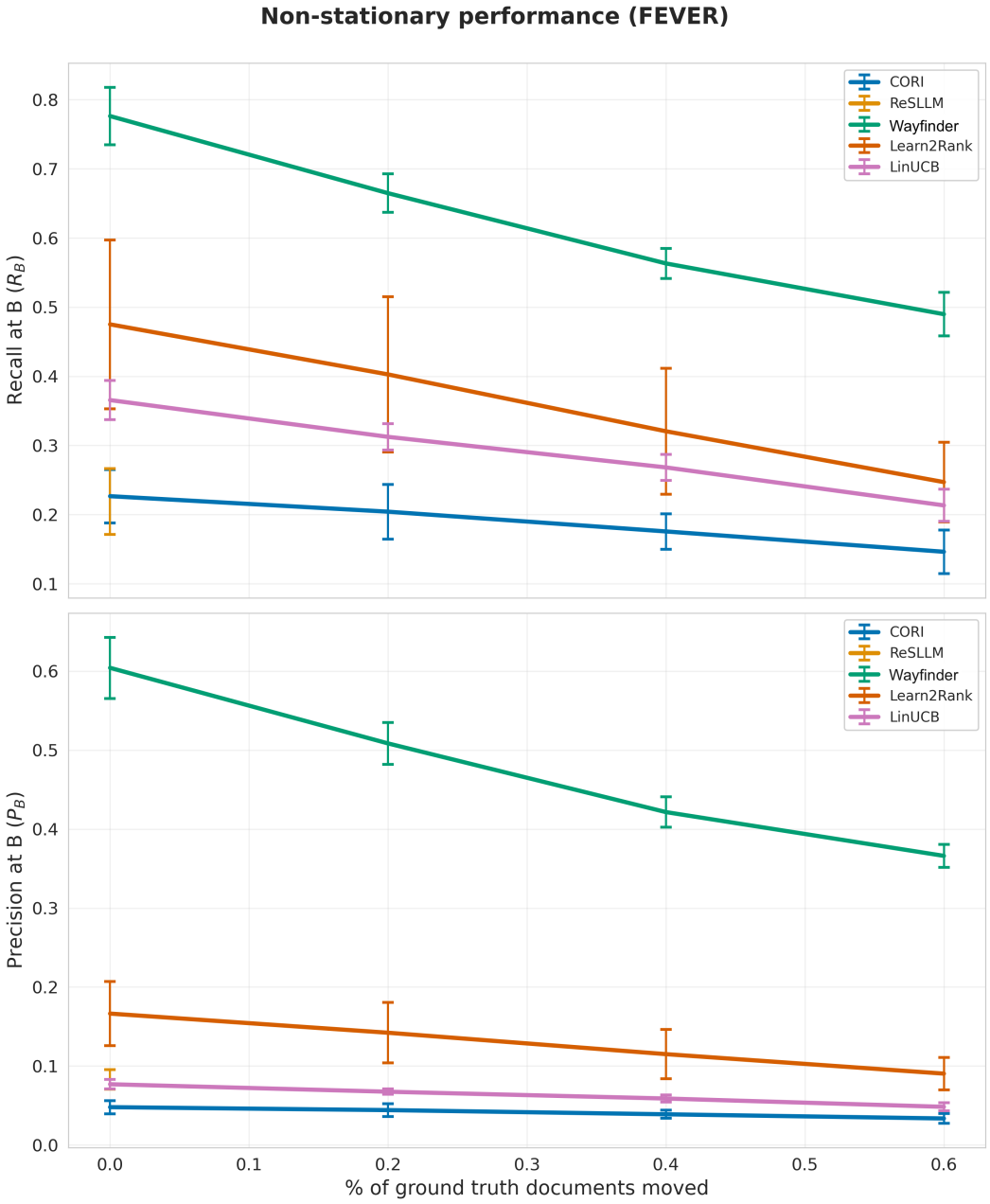


Figure 2: Routing robustness under increasing levels of exogenous changes measured by Precision at B and Recall at B with B=5. As a larger fraction of ground-truth documents is relocated across resources at test time, all methods degrade, with Wayfinder exhibiting better adaptation due to the agent feedback loop.

```
... # omitted for brevity
```

```
<think> Okay, let's tackle this. The user is asking about
the electric chair being an optional form of execution in
specific states as of 2014. First, I need to figure out
which resources might have this information. The
search_past_interactions tool was called with the query
"electric chair execution states 2014", and the response
included several documents.
```

```
Looking at the documents from resource 34, there's a
mention of \"Hill v. McDonough,\" which is a Supreme
Court case about lethal injection in Florida.
That's related to execution methods. Also, \"Capital
punishment\" and \"Capital punishment in Bulgaria\"
are in the list, but Bulgaria isn't one of the states
mentioned. However, the user's query includes Florida,
so maybe the Hill v. McDonough case is relevant.

Other resources like 51 have \"Shapiro v. McManus\"
... # omitted for brevity

Since the search results don't have exact matches, but the
user's query is about execution methods in specific states,
the most relevant resource might be 34, which includes
capital punishment-related documents. Even though it
doesn't directly answer the question, it's the closest.
... # omitted for brevity

Wait, the user's query is about 2014, and the
Hill v. McDonough case was in 2015. So that's slightly
after the user's timeframe. But maybe the information about
Florida's execution methods is still relevant. Since the
search results don't have exact matches, but the user needs
an answer, I should select the resource that has the most
related documents, even if it's not perfect. Resource 34
seems the best fit here.</think>
```

Although none of the retrieved memory samples explicitly state the queried fact about the electric chair, Resource 34, the ground truth resource, exhibits a consistent thematic focus on capital punishment and execution methods, including legal cases and policy discussions relevant to U.S. states. By prioritizing resources with conceptually related topics, Wayfinder is able to prioritize relevant resources for the primary agent. This example demonstrates how Wayfinder performs robust routing under incomplete memory by inferring where relevant information is likely to reside, rather than relying on exact matches or exhaustive search.

C Appendix: Data Construction Process

We construct synthetic federated retrieval environments from BEIR benchmarks to enable controlled evaluation of routing under non-stationarity. Starting from a standard BEIR dataset (corpus, queries, and relevance judgments), we transform static collections into dynamic, multi-resource environments with temporal structure and exogenous drift.

For datasets with graded relevance (e.g., NFCorpus), we retain only highly relevant documents and normalize relevance to binary. Queries without relevant documents after filtering are removed to ensure well-defined routing supervision.

Each dataset is partitioned into multiple independent environments, representing distinct information spaces. Documents are disjointly assigned to environments. Within each environment, documents are organized into a fixed number of resources. Resources are constructed by clustering document embeddings (using a sentence transformer) via a Gaussian Mixture Model, ensuring semantic coherence within each resource and mimic similar content across resources.

To support description-based baselines, each resource is associated with an automatically generated natural-language description. Descriptions are produced by prompting a large language model with representative document samples from the resource (100).

Within each environment, queries are temporally ordered in a query stream to form a single timeline, simulating interactions in a shared environment.

Queries are split temporally into training and test segments using an 80/20 ratio. When enabled, exogenous drift events are realized by randomly moving documents across resources. Drifts are applied only during the test period and preferentially target ground-truth relevant documents to mimic meaningful drifts. Drift is introduced incrementally at multiple rates, where higher drift levels subsume lower ones.

Before drift, we compute ground-truth coverage statistics per environment to characterize task difficulty. The final dataset includes the initial resource state, persona query streams, relevance judgments, train–test splits, pre-computed embeddings, and drift event specifications. All stochastic components are seed-controlled to ensure reproducibility.

This construction enables systematic evaluation of routing policies under controlled resource fragmentation, partial observability, and increasing degrees of non-stationarity.

Tech Startup (cross-corpus) construction. The Tech Startup benchmark is built outside the BEIR pipeline, by combining five distinct real corpora into a single five-resource environment template that mirrors a typical workplace stack. The five resources are sourced as follows: *emails* from Enron [20], *chat channels* from the Alpaca instruction-tuning corpus [29] (response = document, instruction-derived rephrasing = query), *issue tracker* from SWE-bench's GitHub issues [19], *wiki pages* from HotpotQA Wikipedia paragraphs [38], and *code repository* from CodeSearchNet [18]. We assemble five environments using a connected-component partition over (document, query) pairs to ensure that no document or query appears in more than one environment.

Query rewriting and leakage control. Each query is individually rewritten by an LLM rewriter under explicit constraints: (i) reference document *content* rather than the originating application, (ii) avoid surface words that name resources ("email", "Slack", "issue", "ticket", "commit", "pull request", "wiki", etc.), and (iii) read as natural questions a colleague would ask. After rewriting we apply a regex-based blocklist filter that rejects any query still containing resource-revealing tokens; matched queries are re-rewritten until the filter passes. We do not claim that residual style or length artifacts are eliminated, since the underlying corpora differ in discourse type by construction (cross-corpus heterogeneity is the property we want to measure); however, the same artifacts are equally available to every baseline, and the cross-corpus regime's natural difficulty is what makes the supervised classifier (Learn2Rank) competitive in Table 2. Every retained query passes a semantic-retrieval quality gate that requires its ground-truth document to appear in the top-5 of dense retrieval (all-MiniLM-L6-v2) over the environment's documents. The filter removes 4.0% of candidate queries; the surviving $1{,}127$ queries form the released benchmark.

D Trajectory Generation and Citation Statistics

To isolate routing differences from stochastic exploration variance, we pre-compute primary-agent trajectories once per environment and reuse them across Wayfinder, the bandit baselines, and all ablations. This protocol fixes downstream interaction evidence so that performance differences reflect how each method interprets and updates from the same retrieval traces, rather than differences in agent exploration. Trajectories are generated with a production-grade tool-using agent (GPT-5-mini[3]) configured with the structured `search_resources` and `synthesize_answer` tools described in Section 4.1, with a per-rollout step budget of 5 and a context length of 16K tokens.

Citation rate and accuracy. For each trajectory we measure the citation rate $\Pr_t[\mathcal{C}_t \neq \emptyset]$ and the citation accuracy $\mathbb{E}_t[\,|\rho(\mathcal{C}_t) \cap \bar{\mathcal{R}}_t|/|\rho(\mathcal{C}_t)| \mid \mathcal{C}_t \neq \emptyset\,]$. On FEVER, citation rates are high but vary across environments (mean 85.7%, median 90.1%) while citation accuracy is stable (mean 71.3%, std. 4.9%). On NFCorpus, both rate (mean 74.3%) and accuracy (mean 73.1%) are stable. On Tech Startup, citation rate is consistently high (mean $\sim$$95\%$) and accuracy is high (mean $\sim$$73\%$) across the five resource types. Low citation rates yield fewer memory updates rather than noisier ones, supporting the observation that the citation gate trades coverage for precision, and motivating the noise-robustness analysis in Section 7.

[3] `https://platform.openai.com/docs/models/gpt-5-mini`

E Training and Inference Hyperparameters

Routing policy. Wayfinder is implemented as a LoRA-adapted routing policy on top of a Qwen3-8B base model and trained using Group Relative Policy Optimization (GRPO). LoRA adapters use rank 16 and are optimized with a learning rate of 5×10^{-6} and gradient accumulation over 4 steps. Meta-training is performed for 25 GRPO iterations; each iteration samples 8 queries and generates 4 rollouts per query (GRPO group size for advantage normalization). Queries whose group has zero-variance reward (e.g., all rollouts succeed or all fail) are filtered out by GRPO's advantage normalization and contribute no gradient signal.

Rollout and decoding settings. Rollouts are generated with a maximum of 5 interaction turns and a context length of up to 16K tokens. Decoding uses temperature 0.7 and nucleus sampling with $p = 0.95$.

Memory and retrieval. Unless otherwise stated, the routing agent maintains a per-resource memory budget of $M = 100$ documents. At each routing decision, the agent retrieves the top 25 memory entries via semantic search to constrain context length; this retrieval limit is independent of the per-resource memory budget. Memory updates are enabled incrementally at test time and are gated on agent-produced citations.

Embedding model. All embedding-based components across methods use the same frozen sentence encoder, sentence-transformers/all-MiniLM-L6-v2.

Bandit baseline. LinUCB-D is a discounted linear UCB bandit with one arm per resource. Each arm is represented by the centroid of the embeddings of up to 100 documents sampled from the resource, and the context for query q and resource r is a fixed random projection (dimension 32) of the elementwise product of their embeddings, concatenated with their cosine similarity and a bias term (d=34). It selects the top-B resources by UCB score with exploration α=1.0, ridge regularization λ=1.0 and discount γ=0.999. In the closed loop, after the agent acts on its recommendation, each selected resource receives reward 1 if the agent cited a document from it and 0 otherwise.

F Additional Ablations

This appendix collects further Wayfinder ablations that do not fit in the main paper.

F.1 Closed-loop validation

Table 2 is measured closed-loop. Here we compare it with the offline protocol, which replays the same pre-computed primary-agent trajectories for every method so that routing differences are isolated from the agent's exploration variance. Offline, Wayfinder's memory is updated from the citations in those pre-computed trajectories; closed-loop, it is updated from the citations the agent emits after acting on Wayfinder's own recommendations. Table 5 reports both.

Table 5: Closed-loop vs. offline Wayfinder.

	Offline			Closed-loop		
Dataset	$F1_r$	$F1_c$	Saved %	$F1_r$	$F1_c$	Saved %
NFCorpus	50.69	20.05	64.8	47.67	17.10	61.2
FEVER	67.94	59.70	79.3	64.72	50.88	73.7
Tech Startup	83.39	77.66	55.8	86.60	80.78	60.4

$F1_r$: routing F1 at budget B (Appendix G). $F1_c$: the agent's citation F1 against the relevance judgments. Saved %: reduction in the agent's search calls relative to no routing. Closed-loop values are Wayfinder's row of Table 2.

Closed-loop routing F1 is within 3.3 points of offline on every dataset: 3.0 and 3.2 points lower on NFCorpus and FEVER, and 3.2 points higher on Tech Startup. Citation F1 and Saved % move in the same direction, with the largest change in FEVER citation F1 (59.70 to 50.88). Wayfinder's advantage over the baselines in Table 2 therefore does not depend on replaying pre-computed trajectories.

Scope and fairness. All routers in Table 2 run under the same closed-loop protocol: the same primary agent, query stream, routing budget and hard-routed replay. Static routers have no online state; LinUCB-D and Wayfinder are updated from the agent's citations as the stream is processed. We do not report latency.

F.2 Robustness to citation noise

A key claim of Wayfinder is that gating memory updates on agent citations rejects retrieval noise without requiring labels. We stress-test this gate on Tech Startup by perturbing the cited-document field of every trajectory before it reaches the router: *drops* remove a fraction of citations entirely; *flips* replace a fraction's resource provenance with a uniformly chosen incorrect resource. We additionally compare against an *ungated* variant that updates memory from all retrieved documents per trajectory rather than only the cited ones. Each rate is run with a fixed seed. Table 6 reports the outcome.

Table 6: Citation-noise stress test on Tech Startup.

Variant	0%	10%	25%	50%	ungated
Wayfinder (drop)	81.18	85.22	82.64	86.71	–
Wayfinder (flip)	81.98	64.48	46.84	36.48	–
no citation gate	–	–	–	–	84.50

B=1; values are F_1. *drop* removes citations; *flip* rewrites a citation's resource provenance to an incorrect resource; *ungated* updates memory from all retrieved documents.

Drops barely move F_1 (range 81–87): the gate already filters what dropping would discard, so the retained citations remain a high-precision signal. Flips produce a clean monotonic decline ($82 \to 64 \to 47 \to 36$), graceful but proportional degradation when the gate's accuracy itself is corrupted. The ungated baseline (84.50) sits slightly above gated drop 0% (81.18): on Tech Startup, where the five resources come from semantically disjoint corpora, retrievals are already high-precision and the gate's filtering removes mildly useful corroborating evidence. The gate's value is therefore robustness: the contrast between a flat drop curve and a steep flip curve isolates the failure mode Wayfinder can recover from from the one it cannot, plus reduced storage from cited-only memory updates.

F.3 Memory Design (mem0 comparison)

We compare Wayfinder's environment-structured memory against a generic semantic-memory baseline implemented with mem0[4]. The mem0 baseline stores past query–resource trajectories as text, embeds them with a sentence-transformer model, and retrieves semantically similar trajectories by vector similarity; candidate resources are then extracted from the retrieved trajectories. Wayfinder's memory instead represents resources as persistent entities and records interaction outcomes per resource, enabling reasoning about resource-specific reliability over time. Both variants share the same base model, resource representations, routing budget, and training procedure; only the memory structure differs.

Table 7: Generic vs. environment-structured memory.

	NFCorpus			FEVER		
Router	$R_B \uparrow$	$P_B \uparrow$	$F1_B \uparrow$	$R_B \uparrow$	$P_B \uparrow$	$F1_B \uparrow$
w/ mem0	51.04	44.41	47.49	75.32	59.35	66.39
w/ env. mem.	**53.93**	**47.82**	**50.69**	**77.62**	**60.41**	**67.94**

Comparison between generic semantic memory (mem0) and Wayfinder's environment-structured memory. Results are averaged over k folds of environment-level test splits.

Table 7 shows that environment-structured memory outperforms mem0 on both datasets. mem0's flat semantic store mixes exploratory tool calls, intermediate reasoning, and retrieved snippets with

[4] `https://mem0.ai/`

highly variable signal-to-noise, so credit assignment is diffuse. Wayfinder instead maintains resource-specific sketches updated only when the agent produces citations, yielding a higher-precision feedback channel and a stable per-resource representation of "what this resource is useful for."

F.4 Memory Size Sensitivity

We study Wayfinder's sensitivity to per-resource memory capacity by varying the maximum number of retained documents from 5 to 100 while holding the routing policy, reward, and training procedure fixed.

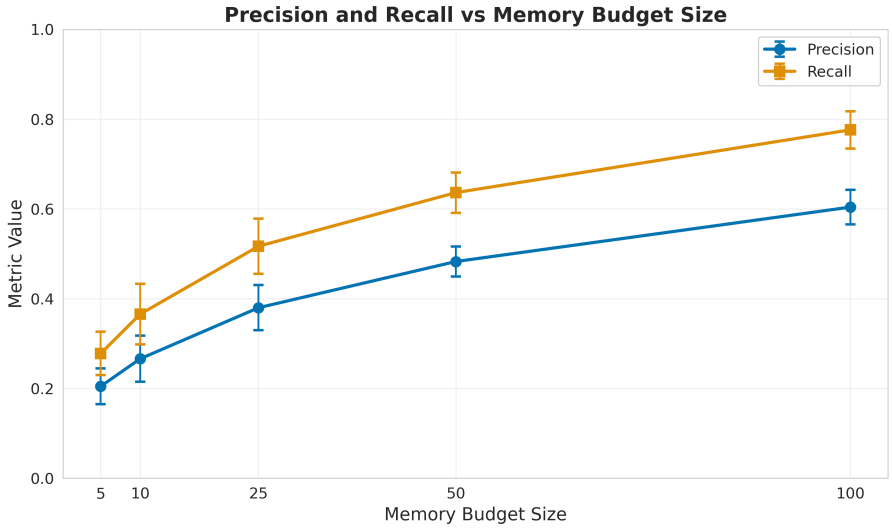


Figure 3: Precision and Recall at B as a function of the per-resource memory budget. Gains diminish beyond moderate memory sizes, indicating that most useful routing signal is captured with limited historical context.

Figure 3 shows that both precision and recall improve with memory capacity, but with diminishing returns beyond moderate sizes, supporting the practicality of Wayfinder under realistic memory budgets.

G Metric Definitions

This appendix provides formal definitions of the evaluation metrics used in the experimental section. Following standard practice in federated search and collection selection [13, 28], we assign each resource $r \in \mathcal{R}$ a nonnegative relevance merit for request q_t based on document-level evidence:

$$m_t(r) \triangleq \sum_{d \in \mathrm{Top}L(r,q_t)} g(\mathrm{rel}(q_t, d)), \tag{4}$$

where $\mathrm{Top}L(r, q_t)$ denotes the top-L retrieved items from resource r, $\mathrm{rel}(q_t, d)$ is a judged or verifier-derived relevance label, and $g(\cdot)$ is a fixed gain function (e.g., $g(\ell) = 2^\ell - 1$ for graded relevance). The merit function is used only to rank relevant resources when constructing $\mathcal{R}^\star(q_t)$.

Recall at B (R_B) Given a routing budget B, Recall at B (R_B) measures the fraction of relevant resources recovered by the routed set:

$$R_B(q_t) \triangleq \frac{|a_t \cap \mathcal{R}^\star(q_t)|}{|\mathcal{R}^\star(q_t)|}, \tag{5}$$

where $a_t \subseteq \mathcal{R}$ denotes the set of resources selected for query q_t (truncated to size B when necessary), and $\mathcal{R}^\star(q_t)$ is the set of up to B ground-truth relevant resources for q_t with the highest merit. This metric evaluates how effectively the router identifies relevant resources under a fixed selection budget and is standard in federated search evaluation [13, 28].

Precision at B (P_B) measures the fraction of selected resources that are relevant:

$$P_B(q_t) \triangleq \frac{|a_t \cap \mathcal{R}^\star(q_t)|}{|a_t|}, \tag{6}$$

with $P_B(q_t) = 0$ when $|a_t| = 0$. This metric quantifies over-selection and complements recall-oriented measures by penalizing the inclusion of irrelevant resources.

$F1$ at B ($F1_B$) summarizes the trade-off between selecting relevant resources and avoiding irrelevant ones, we report the harmonic mean of P_B and R_B:

$$F1_B(q_t) \triangleq \frac{2 \cdot P_B(q_t) \cdot R_B(q_t)}{P_B(q_t) + R_B(q_t)}, \tag{7}$$

with $F1_B(q_t) = 0$ when both $P_B(q_t)$ and $R_B(q_t)$ are zero. $F1_B$ provides a single summary measure of routing quality that balances effectiveness and noise under a fixed resource-selection budget.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction claim the best routing on the intra-corpus benchmarks and that description-based tool retrieval suffices on Tech Startup; Section 7 and Table 2 report these results. **TODO:** recheck after the final intro and abstract edits.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 9 (Limitations and future work) discusses dependence on citations, citation errors, the controlled testbeds, and baselines not compared.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper has no theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 6 and Appendix E give the protocol, models and hyperparameters; Appendix G defines the metrics; the datasets will be released.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The dataset is released at `https://huggingface.co/datasets/miguelrc001/wayfinder`; the code will not be released.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 6 and Appendix E give the data splits, folds, training settings and evaluation protocol.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We report means over k environment-level folds (k=2 for NFCorpus, k=4 for FEVER and Tech Startup) without error bars.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Training and serving the Qwen3-8B router used 8× H100 80GB GPUs; the primary agents are accessed through the OpenAI API (GPT-4o-mini, GPT-5-mini). **TODO:** GPU-hours per adapter and total API cost.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses public datasets and conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: **TODO:** e.g., positive: fewer redundant searches for deployed agents; negative: the router-side store of cited documents concentrates user content and needs access-control and retention policies.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We release no models or data with a high risk of misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets and models are cited. **TODO:** verify each license: NFCorpus, FEVER (CC BY-SA 3.0), Enron emails (public), Alpaca (CC BY-NC 4.0), SWE-bench (MIT), HotpotQA (CC BY-SA 4.0), CodeSearchNet, all-MiniLM-L6-v2 (Apache 2.0), Qwen3-8B (Apache 2.0).

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We release the three testbeds (NFCorpus, FEVER and Tech Startup environments) with documentation of their construction (Section 5, Appendix). **TODO:** documentation at release.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper involves no crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: The paper involves no research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: **[TODO]**

    Justification: LLMs are core components of the method: the router is a LoRA-adapted Qwen3-8B, and the primary agents are GPT-4o-mini and GPT-5-mini (Section 6).

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.