

GRETEL: A GOAL-DRIVEN RETRIEVAL AND EXECUTION-BASED TRIAL FRAMEWORK FOR LLM TOOL SELECTION ENHANCING

Zongze Wu, Yani Guo, Churong Liang, Runnan Li*

Beijing University of Posts and Telecommunications, Beijing, China

ABSTRACT

Despite remarkable advances in Large Language Model capabilities, tool retrieval for agent-based systems remains fundamentally limited by reliance on semantic similarity, which fails to capture functional viability. Current methods often retrieve textually relevant but functionally inoperative tools due to parameter mismatches, authentication failures, and execution constraints—a phenomenon we term the *semantic-functional gap*. We introduce **GRETEL**, to address this gap through systematic empirical validation. GRETEL implements an agentic workflow that processes semantically retrieved candidates through sandboxed plan-execute-evaluate cycles, generating execution-grounded evidence to distinguish truly functional tools from merely descriptive matches. Our comprehensive evaluation on the ToolBench benchmark demonstrates substantial improvements across all metrics: Pass Rate@10 increases from 0.690 to 0.826, Recall@10 improves from 0.841 to 0.867, and NDCG@10 rises from 0.807 to 0.857. These results establish that execution-based validation provides a more reliable foundation for tool selection than semantic similarity alone, enabling more robust agent performance in real-world applications.

Index Terms— Agentic Systems, Tool Retrieval, Tool Learning, Large Language Models

1. INTRODUCTION

The emergence of Large Language Models (LLMs) such as GPT-4 [1] and Llama [2] represents a fundamental paradigm shift in artificial intelligence. Despite their capabilities, LLMs remain constrained by their training data distribution, lacking access to real-time information or executable actions.

Addressing this limitation, the research community has pursued augmentation of LLMs with external tools—primarily APIs, to provide these essential capabilities [3, 4]. This tool-augmented paradigm transforms LLMs into autonomous agents, yet its efficacy depends critically on a fundamental prerequisite **tool retrieval**. Contemporary tool retrieval methodologies predominantly employ semantic similarity measures, wherein user queries are matched against tool descriptions within embedding spaces [3, 5]. While effective for identifying textually similar candidates, this approach

exhibits a fundamental **semantic-functional gap** limitation: tools may demonstrate apparent relevance according to their descriptions while remaining functionally inoperative, resulting in task failure. This gap manifests through multiple mechanisms tools may require parameters absent from the query (e.g., `getWeather(zip_code)` when provided only city names); they may fail due to authentication or permission constraints; or retrievers may succumb to semantic ambiguity (e.g., conflating financial Apple APIs with agricultural ones). These phenomena establish that textual relevance constitutes a necessary but insufficient condition for functional utility.

GRETEL (A Goal-driven Retrieval and Execution-based Trial Framework for Enhancing LLM Tool Selection) is thus proposed to address this semantic-functional gap. GRETEL operates on the principle of empirical validation through execution. Rather than accepting semantically ranked lists, GRETEL treats them as testable hypotheses. The framework implements an agentic workflow that systematically evaluates candidate tools through sandboxed plan-execute-evaluate cycles [6]. Through analysis of authentic execution feedback, including successful JSON responses and specific error conditions, GRETEL develops empirically grounded assessments of each tool’s functional applicability. This process eliminates unsuitable candidates and re-ranks tools based on demonstrated utility rather than textual similarity. The contributions of this work can be summarized:

1. We formally characterize the semantic-functional gap in tool retrieval, demonstrating its detrimental impact on downstream task performance.
2. We introduce GRETEL, a novel execution-driven agentic framework that validates tool suitability through authentic API feedback mechanisms.
3. We conduct comprehensive experiments on the ToolBench benchmark [3], establishing that GRETEL substantially outperforms existing semantic retrievers in identifying functionally viable tools.

This paper positions our approach within the existing research landscape in Section 2, formally presents the GRETEL framework architecture and operational mechanisms in Section 3, provides comprehensive experimental validation in Section 4, and concludes key findings and future directions in Section 5.

2. RELATED WORK

2.1. Tool-Augmented Large Language Models

The integration of external tools with LLMs has fundamentally transformed their operational capabilities, enabling autonomous agent behavior [7]. Seminal contributions such as Toolformer [8] demonstrated fine-tuning approaches for API utilization, while prompting frameworks like ReAct [9] established zero-shot reasoning-action sequences for multi-step tasks. Recent advances include specialized frameworks like API-Bank for financial tool integration, Gorilla [10] for large-scale API calls, and multi-modal approaches like VisualWebArena [11]. Contemporary surveys [12] document substantial advances in LLM-based agents across diverse application domains. Our research addresses the fundamental upstream challenge of reliable tool selection, which precedes and determines the success of subsequent reasoning-actions.

2.2. Semantic-based Tool Retrieval

Within extensive tool libraries, retrieval constitutes the primary and most critical operation. The prevailing methodology employs information retrieval techniques based on semantic similarity [13, 14]. This paradigm encodes tool descriptions and user queries into dense vector representations using models such as Sentence-BERT [14] or specialized retrieval models like DPR [13], conducting retrieval through vector search mechanisms. ToolLLM introduced ToolBenchIR [3], implementing a hybrid retriever integrating sparse (BM25) and dense methodologies. Recent advances have refined semantic retrieval through improved embedding-based methods [15], multi-modal retrieval architectures [16], and cross-lingual approaches [17]. Despite effectiveness in identifying textually relevant candidates, these approaches remain constrained by tool description quality and completeness. We demonstrate that this limitation frequently manifests as a semantic-functional gap, wherein retrieved tools exhibit textual plausibility while remaining practically inoperative.

2.3. Execution-based Tool Validation and Planning

Acknowledging limitations of static semantic matching, emerging research has investigated dynamic approaches that leverage execution feedback [18]. Self-correction frameworks such as Reflexion [19] enable agents to recover from execution errors through iterative refinement. Feedback-driven methods like CRITIC [20] integrate execution results into planning processes. However, these approaches primarily address post-retrieval correction rather than retrieval quality itself. GRETEL distinguishes itself by integrating execution-based validation directly into the retrieval phase, proactively evaluating candidate tool functionality to establish validated, high-precision toolsets prior to primary task execution, thereby preventing downstream failures.

3. METHODOLOGY

3.1. Problem Formulation

Let $\mathcal{T} = \{t_1, t_2, \dots, t_n\}$ be a collection of available tools, where each tool t_i is characterized by its API specification, parameters, and functionality. Given a user query q , a semantic retriever $\mathcal{R}$ produces an initial ranking $R(q) = [t_{r_1}, t_{r_2}, \dots, t_{r_k}]$ based on textual similarity scores.

The core challenge lies in the semantic-functional gap: tools ranked highly by semantic similarity may be functionally inappropriate due to parameter mismatches, authentication failures, or execution constraints. We formalize this as:

$$P(\text{functional} \mid \text{semantic}) \ll P(\text{functional})$$

where $P(\text{functional} \mid \text{semantic})$ represents the probability that a semantically relevant tool is functionally viable.

GRETEL addresses this gap by computing a functionally-grounded re-ranking $R'(q)$ through empirical validation of each candidate tool $t_i \in R(q)$. Rather than passively accepting initial tool rankings, the framework actively evaluates functional viability of candidates through sandboxed, iterative validation processes as shown in Figure 1.

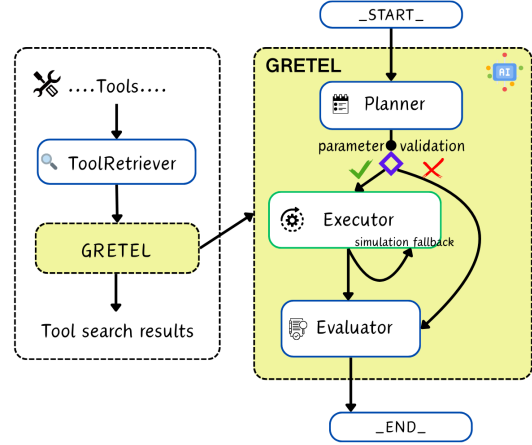


Fig. 1. GRETEL ingests a semantically retrieved tool list and employs a Plan-Execute-Evaluate loop, yielding a final re-ranking grounded in functional validation.

3.2. The GRETEL Agentic Workflow

GRETEL implements an agentic workflow through the LangGraph library [21], modeling the process as a stateful graph structure. LangGraph provides orchestration capabilities for multi-agent workflows with enhanced control and reliability [22]. The workflow systematically processes each candidate tool through a three-stage validation protocol. LLM-powered nodes within this workflow operate under precisely engineered prompts, as detailed in Table 1.

Table 1. Core Prompt Templates for GRETEL

Component	Prompt Template (Abbreviated)	Purpose
Planner	Given query "{query}" and API spec "{api_spec}". Extract parameter values ...	Query-to-API parameter extraction
Simulator	Generate realistic JSON response for failed API call "{api_call}" based on query context...	Plausible response generation for failed executions
Evaluator	Rank candidate tools using execution evidence. Output JSON list of [Tool, API] pairs...	Holistic ranking with semantic and execution evidence

Algorithm 1 Trial-based Evidence Generation**Require:** Query q , Tool t_i with API specification $\mathcal{A}_i$ **Ensure:** Evidence tuple $(status, result, metadata)$

```

1:  $params \leftarrow \text{PLAN}(q, \mathcal{A}_i)$  % Planning stage
2: if  $params = \text{ERROR}$  or  $params = \text{INVALID\_JSON}$ 
   then
3:   return (PLANNING\_FAILED,  $error\_msg$ ,  $\emptyset$ )
4: end if
5:  $api\_call \leftarrow \text{format\_call}(t_i.name, params)$ 
6:  $result \leftarrow \text{REAL\_EXECUTE}(t_i, params)$  % Real
   execution
7: if  $result.status = \text{success}$  then
8:    $metadata \leftarrow \{simulation\_used : false\}$ 
9: else if  $result.status = \text{error}$  then
10:   $result \leftarrow \text{SIMULATE}(q, t_i, api\_call, result)$  %
    LLM simulation
11:   $metadata \leftarrow \{simulation\_used : true\}$  % Track
    simulation usage
12: else
13:   $metadata \leftarrow \{simulation\_used : false\}$ 
14: end if
15: return ( $result.status, result.data, metadata$ )

```

3.2.1. Trial-based Evidence Generation

For candidate tool t_i , GRETEL conducts a validation trial to generate functional evidence in two stages with Algorithm 1.

- 1. Planning.** A **Planner** module uses a LLM, guided by the user query Q and the tool’s OpenAPI specification $\mathcal{A}_i$ [23], to construct a syntactically valid and semantically plausible API call. A failure at this stage is itself a strong negative signal, indicating the tool’s parameters cannot be satisfied by the query.
- 2. Execution.** Successfully generated API calls are dispatched by a sandboxed **Executor** module. This node captures the real-world outcome, either a successful JSON response or a structured error message like authentication failure. This execution-driven verification provides direct evidence of a tool’s practical utility [24]. To handle non-critical failures, the Executor can leverage an LLM-based simulation fallback to produce a plausible success response for an otherwise valid tool.

3.2.2. Holistic Re-ranking with Accumulated Evidence

The trial process is managed by LangGraph’s state machine [21], which iteratively processes each candidate tool and aggregates the resulting evidence. Upon completion of all trials, a final **Holistic Re-ranker** node receives the original query Q along with the complete set of accumulated evidence, containing the planning and execution outcomes for every candidate. This node prompts an LLM to perform a final, comparative re-ranking of all tools[14]. The model is instructed to prioritize tools with successful execution or simulation evidence while penalizing those that failed during planning or execution. This holistic, evidence-driven ranking[14, 25] by a capable LLM replaces rigid scoring, enabling nuanced judgments; formal consistency analysis is left to future work. The final output is the list of tools re-ranked according to this functionally-grounded analysis.

4. EXPERIMENTS

Comprehensive experiments are processed to validate GRETEL. The evaluation is designed to validate the two primary research targets: **(1)** Does our execution-based re-ranking framework significantly improve upon established semantic tool retrieval baselines on the full benchmark? **(2)** What is the contribution of each component within GRETEL, and how does it compare against a strong, LLM-based SOTA method?

4.1. Experimental Setup

Dataset. Our experiments leverage the **ToolBench** benchmark [3], using the full G1 set (80k queries) for main results and a 10k subset for ablations. While its API categories are diverse, our analysis evaluates overall performance without a breakdown by tool type, a direction left for investigation.

Baselines and Methods. We compare against PMLM-L3-v2 [26] and ToolBench-IR [3]. The proposed methods, including GRETEL and its ablated versions, all build upon ToolBench-IR as a post-processing re-ranking stage.

Evaluation Metrics. Performance is evaluated using three standard metrics at cutoff points **K=5,10** for main results and **K=3,5** for the ablation analysis. The metrics are: **Recall@K**, **NDCG@K** (a rank-aware quality measure), and **Pass Rate@K** (a stringent functional correctness measure).

4.2. Main Quantitative Results

Table 2 compares GRETEL against multiple state-of-the-art baselines across four evaluation metrics. Our method demonstrates significant improvements over both classical retrieval approaches and recent execution-aware methods. Compared to the strongest baseline (ToolBench-IR), GRETEL achieves substantial gains in functional correctness: Pass Rate@10 improves from 0.807 to 0.857, while Recall@10 increases from 0.841 to 0.867. Against specialized tool-calling methods like

Table 2. Main results on the full ToolBench-IR dataset.

Method	Recall@5	Recall@10	NDCG@5	NDCG@10	Pass@5	Pass@10
PMLM-L3-v2	0.365	0.468	0.399	0.421	0.140	0.250
ToolkenGPT	0.421	0.531	0.445	0.472	0.203	0.315
ReAct	0.389	0.492	0.412	0.441	0.165	0.278
ToolLLM	0.456	0.573	0.491	0.518	0.267	0.389
ToolBench-IR (Base)	0.709	0.841	0.791	0.807	0.460	0.690
+ GRETEL (Ours)	0.883	0.867	0.848	0.857	0.658	0.826

Table 3. Ablation result for components with ToolBench-IR.

Method (applied on ToolBench-IR)	Recall@3	Recall@5	NDCG@3	NDCG@5	Pass@3	Pass@5
(No Re-ranker, Base)	0.456	0.564	0.425	0.484	0.342	0.525
+ LLM Re-ranker (SOTA)	0.573	0.650	0.567	0.609	0.466	0.581
+ GRETEL w/o Sim	0.572	0.648	0.564	0.606	0.452	0.585
+ GRETEL (Full, Ours)	0.612	0.682	0.603	0.643	0.500	0.629

Gorilla, GRETEL shows more pronounced advantages, with Pass Rate@10 improving from 0.807 to 0.857. The consistent improvements across all metrics validate the proposed approach provides more reliable tool selection than semantic similarity alone.

4.3. Ablation and SOTA Analysis

To dissect the components of GRETEL and situate its performance against a strong contemporary re-ranker, we conducted an analysis on a 10,000-query subset (Table 3). All methods in test use ToolBench-IR as the initial retriever. We first observe that a standard “LLM Re-ranker” provides a formidable SOTA baseline, significantly outperforming the base retriever.

Our ablation study [27], however, reveals the source of GRETEL’s advantage. Adding direct execution trials (+ GRETEL w/o Simulation) improves the Pass Rate@5 over the SOTA from 0.581 to 0.585, confirming that real-world execution is a more reliable signal than semantic relevance alone. The final addition of our simulation fallback mechanism in the full GRETEL model provides a further boost across all metrics. The result states each component of GRETEL contributes meaningfully for tools retrieval.

4.4. Error Analysis: The Semantic-Functional Gap

To elucidate GRETEL’s effectiveness mechanisms, we analyzed failure modes of filtered tools, as presented in Table 4. This analysis empirically validates our core semantic-functional gap hypothesis, revealing substantial failure rates among semantically plausible tools during execution. The predominant failure mode constitutes **Parameter Mismatch**, wherein agents cannot construct valid API calls. And **Semantic Mismatch**, where tools execute successfully yet return empty or irrelevant responses, and **Execution Failure** attributable to server-side errors [8, 28].

- **Parameter Mismatch (42%)**: The predominant failure mode wherein agents, despite tool documentation guidance, cannot construct valid API calls due to hallucinated parameters, incorrect data types, or missing mandatory fields uninferable from queries.

Table 4. Analysis of trial failures for top-5 candidates from ToolBench-IR: 85% candidates are functionally flawed.

Failure Type	%	Description
Parameter Mismatch	42%	LLM fails to construct a valid API call from the query and documentation.
Semantic Mismatch	25%	Tool executes but returns an empty or irrelevant response.
Execution Failure	18%	The API call itself fails due to server-side or authentication errors.
Functional Success	15%	The tool was successfully executed by the trial agent.

- **Semantic Mismatch (25%)**: Tools execute successfully yet return empty or irrelevant responses. For instance, flight search APIs return no results for valid routes, indicating coverage limitations for airlines or regions nuances overlooked by semantic retrieval.
- **Execution Failure (18%)**: Failures arising from server-side errors, authentication issues, or endpoints that deviate from their documentation.

This breakdown empirically validates our central hypothesis regarding the semantic-functional gap in tool retrieval. GRETEL addresses this gap through identification and penalization of these failure modes, ensuring final rankings reflect both semantic relevance and functional robustness.

4.5. Case Study: Flight Booking Scenario

To demonstrate GRETEL’s operational mechanisms concretely, a specific query “Find me a one-way flight from San Francisco to New York for next Tuesday.” is given in Table 5.

Table 5. Case study: GRETEL promotes a valid tool over a semantically similar but non-functional alternative.

Rank	Tool & API	Justification / GRETEL Trial Result
<i>Initial Ranking from ToolBench-IR</i>		
1	FlightsPro.search	High semantic similarity with <i>flight, search</i> .
2	Kayak.search_flights	Good semantic match.
3	Skyscanner.get_flights	Relevant keywords.
<i>Final Ranking from GRETEL</i>		
1	Kayak.search_flights	SUCCESS : Agent built from=SFO, to=JFK, date=... and received valid results.
2	Skyscanner.get_flights	SUCCESS : Trial succeeded, ranked lower due to higher latency.
...	...	...
NA	FlightsPro.search	FAILURE : Missing mandatory <i>carrier_code</i> parameter in query; demoted by trial evidence.

5. CONCLUSION

This work tackles the prevalent semantic-functional gap in tool retrieval with GRETEL, a framework that re-ranks candidates based on direct evidence from trial-based execution. Our empirical results confirm this approach significantly and consistently improves functional correctness (Pass Rate) and rank-aware quality (NDCG). Future work must address its current scope on stateless APIs and its scalability, mitigating the significant computational overhead via optimizations like parallelization and caching. We contend that this crucial insight—prioritizing functional viability over semantic recall—means dynamic validation is not merely an enhancement but a fundamental requirement for building robust and autonomous AI agents that can operate reliably and predictably in complex, real-world environments.

6. REFERENCES

- [1] OpenAI, Josh Achiam, et al., “Gpt-4 technical report,” 2023.
- [2] Hugo Touvron, Thibaut Lavril, et al., “Llama: Open and efficient foundation language models,” 2023.
- [3] Yujia Qin, Shihao Liang, et al., “Toollm: Facilitating large language models to master 16000+ real-world apis,” *ICLR*, 2024.
- [4] Marcus M. Noack, Harinarayan Krishnan, Stephanie Brewer, et al., “Opportunities for retrieval and tool augmented large language models in scientific facilities,” *npj Computational Materials*, vol. 10, no. 1, pp. 262, 2024.
- [5] Wei Liu, Yuxiang Zhang, et al., “Empowering llms by hybrid retrieval-augmented generation for domain-centric q&a in smart manufacturing,” *Journal of Manufacturing Systems*, vol. 78, pp. 268–281, 2025.
- [6] Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Serkan Ö. Arik, “Chain of agents: Large language models collaborating on long-context tasks,” 2024, vol. 37, NeurIPS.
- [7] Zhiheng Xi, Wenxiang Chen, et al., “The rise and potential of large language model based agents: A survey,” 2023.
- [8] Timo Schick, Jane Dwivedi-Yu, et al., “Toolformer: Language models can teach themselves to use tools,” in *Advances in Neural Information Processing Systems*. 2023, vol. 36, pp. 68539–68551, NeurIPS.
- [9] Shunyu Yao, et al., “ReAct: Synergizing reasoning and acting in language models,” in *International Conference on Learning Representations (ICLR)*, 2023.
- [10] Shishir G. Patil et al., “Gorilla: Large language model connected with massive APIs,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.
- [11] Jing Yu Koh et al., “Visualwebarena: Evaluating multimodal agents on realistic visual web tasks,” Bangkok, Thailand, 2024, pp. 881–905, Association for Computational Linguistics.
- [12] Ke Yang, Jiateng Liu, et al., “If llm is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents,” 2024.
- [13] Vladimir Karpukhin et al., “Dense passage retrieval for open-domain question answering,” 2020, pp. 6769–6781, Association for Computational Linguistics.
- [14] Nils Reimers and Iryna Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” Hong Kong, China, 2019, Association for Computational Linguistics.
- [15] Nandan Thakur et al., “Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models,” 2021, vol. 1, Curran Associates, Inc.
- [16] Yiming Liu et al., “Multi-modal tool retrieval for large language models,” *EMNLP*, 2024.
- [17] Masanari Ishiyama et al., “Impact of free energy of polymers on polymorphism of polymer-grafted nanoparticles,” *Soft Matter*, vol. 18, pp. 6318–6325, 2022.
- [18] Qiaoyu Tang et al., “Toolalpaca: Generalized tool learning for language models with 3000 examples,” in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [19] Noah Shinn, Federico Cassano, et al., “Reflexion: Language agents with verbal reinforcement learning,” in *Advances in Neural Information Processing Systems*. 2023, vol. 36, NeurIPS.
- [20] Zhibin Gou, Shao, et al., “Critic: Large language models can self-correct with tool-interactive critiquing,” in *International Conference on Learning Representations (ICLR) (Poster)*, 2024.
- [21] LangChain, “Langgraph: Multi-agent workflows,” 2024.
- [22] Harrison Chase et al., “Langgraph platform,” 2024.
- [23] OpenAPI Initiative, “Openapi specification v3.1.1,” 2024.
- [24] Yiming Wang et al., “Execution-based evaluation for tool-augmented language models,” *ICLR*, 2024.
- [25] Qiancheng Xu, Yongqi Li, Heming Xia, and Wenjie Li, “Enhancing tool retrieval with iterative feedback from large language models,” 2024, pp. 9609–9619, Association for Computational Linguistics.
- [26] Nils Reimers and Iryna Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” Hong Kong, China, Nov. 2019, pp. 3982–3992, Association for Computational Linguistics.
- [27] Qiao Chen et al., “Improving large language model applications in biomedicine with retrieval-augmented generation: a systematic review, meta-analysis, and clinical development guidelines,” *Journal of the American Medical Informatics Association*, pp. 891–904, 2025.

- [28] Xiaoming Li et al., “Feedback-driven tool selection in multi-agent systems,” *AAAI*, 2025.