Towards Completeness-Oriented Tool Retrieval for Large Language Models

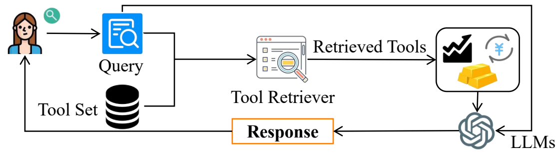
Anonymous ACL submission

Abstract

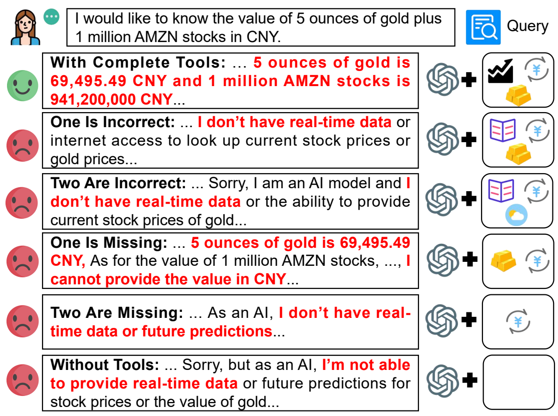
Recently, enhancing the capabilities of Large Language Models (LLMs) through interaction with external tools has gathered widespread interest, where tool retrieval emerges as a crucial step. Existing tool retrieval approaches only focus on semantic matching. However, effective tool retrieval requires consideration of collaborative invocation among multiple tools rather than solely evaluating the utility of individual tools, which presents a challenge to existing tool retrieval methods. To address this, we propose a novel **CO**llaborative **L**earning-based **T**ool Retrieval approach, **COLT**, which manages not only the semantic matching between user queries and tool descriptions but also takes into account the collaborative information of tools. Extensive experiments on both the open benchmark and the introduced TOOLLENS dataset show that COLT achieves superior performance. Notably, the performance of BERT-mini (11M) with our COLT framework outperforms BERT-large (340M), which has 30 times more parameters. Our codes and data are publicly available at `https://anonymous.4open.science/r/COLT-4D13`.

1 Introduction

The integration of tool learning into large language models (LLMs) has emerged as a groundbreaking advancement (Schick et al., 2023; Parisi et al., 2022; Li et al., 2023; Ye et al., 2024), facilitating access to real-time data and the execution of complex computations. By integrating tool learning, LLMs transcend the confines of their outdated or limited pre-trained knowledge (Brown et al., 2020), offering responses to user queries with markedly enhanced accuracy and relevance (Huang et al., 2023; Qin et al., 2023b). However, as real-world systems usually have a vast number of tools, it is infeasible to take the descriptions of all tools as input for LLMs due to the length limitations and latency constraints. Thus, as illustrated in Figure 1(a), developing an efficient tool retrieval system becomes essential to fully exploit the potential of tool-augmented LLMs (Gao et al., 2024).



(a) Pipeline of user interaction with tool-augmented LLMs.



(b) Illustration of different response with different tools.

Figure 1: An illustration of tool retrieval for LLMs with tool learning.

Typically, existing tool retrieval approaches directly employ dense retrieval techniques (Qin et al., 2023b; Yuan et al., 2024), solely focusing on matching semantic similarities between queries and tool descriptions. Yet, these approaches fall short when addressing multifaceted queries that require a collaborative effort from multiple tools to formulate a complete response. For instance, in Figure 1(b), consider a user's request to calculate the value of 5 ounces of gold plus 1 million AMZN stocks in CNY. Such a query necessitates the simultaneous use of tools for gold prices, stock values, and currency exchange rates. The absence of any of these tools yields an incomplete answer, underscoring the limitations of dense retrieval methods that overlook the necessity for tools to interact collaboratively. Therefore, ensuring the completeness of the retrieved tools is an essential aspect of a tool retrieval system, which is often neglected by traditional tool retrieval approaches.

Toward this end, this paper proposes **COLT**, a novel **CO**llaborative **L**earning-based **T**ool retrieval approach, aiming at completeness-oriented tool retrieval. In order to capture the intricate collaborative relationship among tools, a concept of scene is proposed to indicate a group of collaborative tools. Based on this, COLT integrates three bipartite graphs among queries, scenes, and tools. More specifically, given the initial semantic embedding from the pre-trained language model, the high-order collaborative relationship is better integrated via the message propagation and cross-view graph contrastive learning among these graphs. To facilitate the concurrent acquisition of a variety of tools from the entire ground-truth set without favoring any specific tool, the learning objective incorporates a list-wise multi-label loss.

Moreover, traditional retrieval metrics like Recall and NDCG (Järvelin and Kekäläinen, 2002) fail to capture the completeness necessary for effective tool retrieval. As illustrated in Figure 1(b), the exclusion of any essential tool from the ground-truth tool set compromises the ability to fully address user queries, indicating that metrics focused solely on individual tool ranking performance do not suffice when multiple tools are required. To bridge this gap, we introduce COMP@K, a new metric designed to assess tool retrieval performance based on completeness, which can serve as a reliable indicator of how well a tool retrieval system for downstream tool learning applications. Additionally, we construct a new dataset called TOOLLENS, in which a query is typically paired with multiple tools, reflecting the multifaceted nature of user requests in real-world scenarios.

To summarize, our main contributions are:

• The collaborative relationships among multiple tools in LLMs have been thoroughly studied, which reveals that incomplete tool retrieval hinders accurate answers, underscoring the integral role each tool plays in the collective functionality.

• We introduce COLT, a novel tool retrieval approach that uses message propagation and cross-view graph contrastive learning among queries, scenes, and tools, incorporating better collaborative information among various tools.

• The extensive experimental results demonstrate the superior performance of COLT against state-of-the-art dense retrieval methods in both tool retrieval and downstream tool learning. Additionally, we release a new dataset and introduce a novel evaluation metric, both of which are tailor-made for assessing multi-tool usage in LLMs.

2 Our Approach: COLT

In this section, we first introduce the formulation of tool retrieval. Then we describe the details of the proposed COLT approach.

2.1 Task Formulation

Formally, given a user query $q \in \mathcal{Q}$, the goal of tool retrieval is to filter out the top-K most suitable tools $\{t^{(1)}, t^{(2)}, \dots, t^{(K)}\}$ from the full tool set $\mathcal{T} = \{(t_1, d_1), (t_2, d_2), \dots (t_N, d_N)\}$, where each element represents a specific tool t_i associated with its description d_i and N is the number of tools in the tool set.

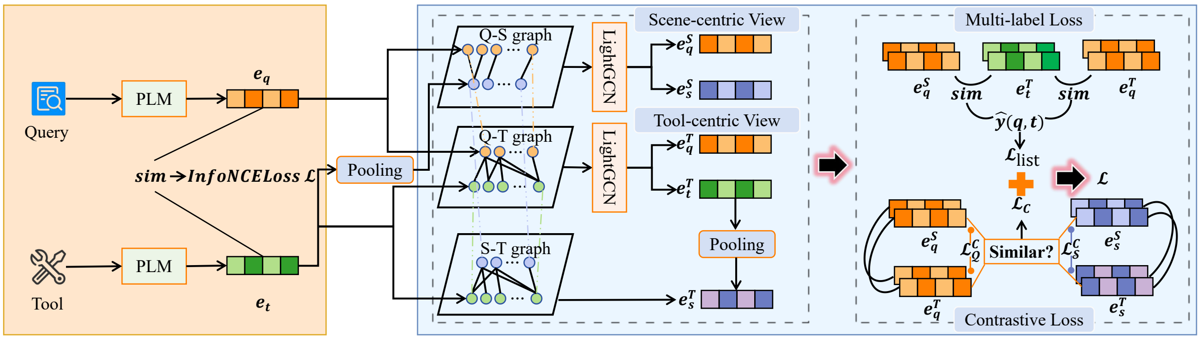
2.2 Overview of COLT

As illustrated in Figure 2, COLT employs a two-stage learning strategy, encompassing semantic learning followed by collaborative learning. In the first phase, the semantic learning module processes both queries and tools to derive their semantic representations, which aims to align these representations closely within the semantic space. Subsequently, the collaborative learning module enhances these preliminary representations by introducing three bipartite graphs among queries, scenes, and tools. Through dual-view graph contrastive learning within these three bipartite graphs, COLT is able to capture the high-order collaborative information between tools. Furthermore, a list-wise multi-label loss is utilized in the learning objective to facilitate the balanced retrieval of diverse tools from the complete ground-truth set, avoiding undue emphasis on any specific tool.

In the following sections, we will present the details of these two key learning stages in COLT.

2.3 Semantic Learning

In the first stage of COLT, we adopt the established dense retrieval (DR) framework (Zhao et al., 2023; Guo et al., 2022), leveraging pre-trained language models (PLM) like BERT (Kenton and Toutanova, 2019) to encode both the query q and tool t into low-dimensional vector. Specifically, we employ a



(a) Semantic Learning (b) Collaborative Learning

Figure 2: The architecture of the proposed two-stage learning framework COLT for tool retrieval.

bi-encoder architecture, with the cosine similarity between the encoded vectors serving as the preliminary relevance score:

$$\widehat{y}_{\text{SL}}(q,t) = \text{sim}(\mathbf{e}_q, \mathbf{e}_t),$$

where $\mathbf{e}_q$ and $\mathbf{e}_t$ denote the mean pooling vectors from the final layer of the chosen PLM, and $\text{sim}(\cdot,\cdot)$ represents the cosine similarity function.

For training, we utilize the InfoNCE loss (Gutmann and Hyvärinen, 2010; Xiong et al., 2020), a standard contrastive learning technique in training DR models, which contrasts positive pairs against negative ones. This semantic learning phase ensures good representations for each query and tool from the text description view. Yet, relying solely on semantic-based retrieval is insufficient for complete tool retrieval, as it often falls short in addressing multifaceted queries effectively.

2.4 Collaborative Learning

Bipartite Graphs in Tool Retrieval. To capture the collaborative information between tools and achieve completeness-oriented tool retrieval, we first formulate the relation between queries and tools with three bipartite graphs. Specifically, we conceptualize the ground-truth tool set for each query as a "scene", considering that a collaborative operation of multiple tools is essential to fully address multifaceted queries. Given the query "I want to travel to Paris.", it doesn't merely seek a single piece of information but initiates a "scene" of travel planning, requiring an array of tools for navigation, weather forecasting, transportation, and accommodation. This scenario underscores the need for scene matching beyond traditional semantic search or recommendation scenarios, where the focus is on selecting any relevant documents or items without considering their collaborative utility. Accordingly, we construct three bipartite graphs linking queries, scenes, and tools, i.e., Q-S (Query-Scene) graph, Q-T (Query-Tool) graph, and S-T (Scene-Tool) graph. By formulating these three graphs, we can further capture the high-order relationships among tools with graph learning, facilitating a scene-based understanding that aligns to achieve a completeness-oriented tool retrieval.

Dual-view Graph Collaborative Learning. Leveraging the initial query and tool representations derived from the first-stage semantic learning, along with the three constructed bipartite graphs, we introduce a dual-view graph collaborative learning framework. This framework is designed to capture the relationships between tools, as depicted in Figure 2 (b). It assesses the relevance between queries and tools from two views:

- **Scene-centric View:** Through the Q-S graph and S-T graph, this view captures the relevance between queries and tools mediated by a scene. This offers a nuanced view that considers the collaborative context in which tools operate together to fulfill a query's requirements.
- **Tool-centric View:** Utilizing the Q-T graph, this view establishes a direct relevance between each query and its corresponding tools, providing a straightforward measure of their relevance.

This dual-view framework allows for comprehensive accessing of query-tool relevance, integrating both direct relevance and the broader context of tool collaboration within scenes, thereby enhancing the completeness of the tool retrieval.

For the scene-centric view, we adopt the simple but effective Graph Neural Network (GNN)-based LightGCN (He et al., 2020) model to delve into the complex relationships between queries and scenes. This is achieved through iterative aggregation of neighboring information across I layers within the Q-S graph. The aggregation process for the i-th layer, enhancing the representations of queries $\mathbf{e}_q^{S(i)}$ and scenes $\mathbf{e}_s^{S(i)}$, is defined as follows:

$$\begin{cases} \mathbf{e}_q^{S(i)} = \sum_{s \in \mathcal{N}_q^S} \frac{1}{\sqrt{|\mathcal{N}_q^S|}\sqrt{|\mathcal{N}_s^Q|}} \mathbf{e}_s^{S(i-1)}, \\ \mathbf{e}_s^{S(i)} = \sum_{q \in \mathcal{N}_s^Q} \frac{1}{\sqrt{|\mathcal{N}_q^S|}\sqrt{|\mathcal{N}_s^Q|}} \mathbf{e}_q^{S(i-1)}, \end{cases} \quad (1)$$

where $\mathcal{N}_q^S, \mathcal{N}_s^Q$ represent the sets of neighbors of query q and scene s in Q-S graph, respectively. $\mathbf{e}_q^{S(0)}$ comes from the representations learned in the first semantic learning stage, while $\mathbf{e}_s^{S(0)}$ is derived from the mean pooling of the representations of ground-truth tools associated with each scene:

$$\mathbf{e}_s^{S(0)} = \frac{1}{|\mathcal{N}_s^T|} \sum_{t \in \mathcal{N}_s^T} \mathbf{e}_t, \quad (2)$$

where $\mathcal{N}_s^T$ represents the set of first-order neighbors of scene s in S-T graph.

Then we sum the representations from the 0-th layer to the I-th layer to get the final query representations $\mathbf{e}_q^S$ and scene representation $\mathbf{e}_s^S$ for the scene-centric view:

$$\begin{cases} \mathbf{e}_q^S = \mathbf{e}_q^{S(0)} + \cdots + \mathbf{e}_q^{S(I)}, \\ \mathbf{e}_s^S = \mathbf{e}_s^{S(0)} + \cdots + \mathbf{e}_s^{S(I)}. \end{cases} \quad (3)$$

In parallel to the scene-centric view, the tool-centric view utilizes LightGCN on the Q-T graph to refine query and tool representations through iterative aggregation. For each layer i, the enhanced representations, $\mathbf{e}_q^{T(i)}$ for queries and $\mathbf{e}_t^{T(i)}$ for tools, are derived as follows:

$$\begin{cases} \mathbf{e}_q^{T(i)} = \sum_{t \in \mathcal{N}_q^T} \frac{1}{\sqrt{|\mathcal{N}_q^T|}\sqrt{|\mathcal{N}_t^Q|}} \mathbf{e}_t^{T(i-1)}, \\ \mathbf{e}_t^{T(i)} = \sum_{q \in \mathcal{N}_t^Q} \frac{1}{\sqrt{|\mathcal{N}_q^T|}\sqrt{|\mathcal{N}_t^Q|}} \mathbf{e}_q^{T(i-1)}, \end{cases} \quad (4)$$

where $\mathcal{N}_q^T, \mathcal{N}_t^Q$ represent neighbors of query q and tool t in Q-T graph, respectively. $\mathbf{e}_q^{T(0)}$ and $\mathbf{e}_t^{T(0)}$ are obtained from the first-stage semantic learning.

Then we sum the representations from the 0-th layer to the I-th layer to derive the final query representations $\mathbf{e}_q^T$ and tool representation $\mathbf{e}_t^T$ for the tool-centric view:

$$\begin{cases} \mathbf{e}_q^T = \mathbf{e}_q^{T(0)} + \cdots + \mathbf{e}_q^{T(I)}, \\ \mathbf{e}_t^T = \mathbf{e}_t^{T(0)} + \cdots + \mathbf{e}_t^{T(I)}. \end{cases} \quad (5)$$

Furthermore, leveraging the learned tool representations $\mathbf{e}_t^T$ and the S-T graph, the scene representation $\mathbf{e}_s^T$ within the tool-centric view can be obtained by pooling all related tool representations:

$$\mathbf{e}_s^T = \frac{1}{|\mathcal{N}_s^T|} \sum_{t \in \mathcal{N}_s^T} \mathbf{e}_t^T. \quad (6)$$

In summary, our dual-view graph collaborative learning framework yields two sets of embeddings: $\mathbf{e}_q^S$ and $\mathbf{e}_s^S$ from the scene-centric view, and $\mathbf{e}_q^T$ and $\mathbf{e}_s^T$ from the tool-centric view, for queries and scenes respectively. Then, the final matching score of each given query-tool pair (q, t) is implemented according to the following formula:

$$\widehat{y}(q, t) = \mathrm{sim}(\mathbf{e}_q^S, \mathbf{e}_t^T) + \mathrm{sim}(\mathbf{e}_q^T, \mathbf{e}_t^T). \quad (7)$$

Learning Objective. To effectively capture high-order collaborative relationships between tools and align the cooperative interactions across two views, we utilize a cross-view contrastive loss. Specifically, the representations of queries and scenes can be learned by optimizing the cross-view InfoNCE (Gutmann and Hyvärinen, 2010) loss:

$$\mathcal{L}_{\mathcal{Q}}^C = -\frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \log \frac{e^{\mathrm{sim}(\mathbf{e}_q^S, \mathbf{e}_q^T)/\tau}}{\sum_{q- \in \mathcal{Q}} e^{\mathrm{sim}(\mathbf{e}_q^S, \mathbf{e}_{q-}^T)/\tau}}, \quad (8)$$

$$\mathcal{L}_{\mathcal{S}}^C = -\frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \log \frac{e^{\mathrm{sim}(\mathbf{e}_s^S, \mathbf{e}_s^T)/\tau}}{\sum_{s- \in \mathcal{S}} e^{\mathrm{sim}(\mathbf{e}_s^S, \mathbf{e}_{s-}^T)/\tau}}, \quad (9)$$

where τ is the temperature parameter.

To ensure the complete retrieval of diverse tools from the full set of ground-truth tools, without favoring any particular tool, we design a list-wise multi-label loss as the main learning objective loss. Given a query q, the labeled training data is $\Gamma_q = \{\mathcal{T}_q = \{t_i, d_i\}, y = \{y(q, t_i)\} | 1 \le i \le L\}$, where $\mathcal{T}_q$ denotes a tool list with length L, comprising N_q ground-truth tools and $L - N_q$ negative tools that are randomly sampled from the entire tool set. $y(q, t_i)$ is the binary relevance label, taking a value of either 0 or 1, and the ideal scoring function should meet the following criteria:

$$p_q^t = \frac{\gamma(y(q, t))}{\sum_{t' \in \mathcal{T}_q} \gamma(y(q, t'))}, \quad (10)$$

where p_q^t is the probability of selecting tool t. $\gamma(y(q, t)) = 1$ if $y(q, t) = 1$ and $\gamma(y(q, t)) = 0$ if $y(q, t) = 0$.

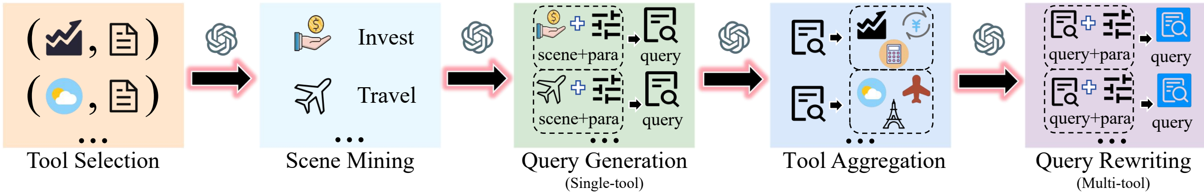


Figure 3: An overview of the dataset construction pipeline of TOOLLENS.

Similarly, given the predicted scores $\{\widehat{y}(q,t_1),\cdots,\widehat{y}(q,t_L)\}$, the probability of selecting tool t can be derived:

$$\widehat{p_q^t} = \frac{\gamma(\widehat{y}(q,t))}{\sum_{t'\in\mathcal{T}_q}\gamma(\widehat{y}(q,t'))}. \tag{11}$$

Therefore, the list-wise multi-label loss function is then formulated to minimize the discrepancy between these two probability distributions:

$$\mathcal{L}_{\text{list}} = -\sum_{q\in\mathcal{Q}}\sum_{t\in\mathcal{T}_q} p_q^t \log \widehat{p_q^t} + (1-p_q^t)\log(1-\widehat{p_q^t}), \tag{12}$$

Based on the multi-label loss $\mathcal{L}_{\text{list}}$ and the contrastive loss $\mathcal{L}_{\mathcal{Q}}^{C}$, the final loss $\mathcal{L}$ for our proposed COLT is formally defined as:

$$\mathcal{L} = \mathcal{L}_{\text{list}} + \lambda(\mathcal{L}_{\mathcal{Q}}^{C} + \mathcal{L}_{\mathcal{S}}^{C}), \tag{13}$$

where λ is the co-efficient to balance the two losses.

3 Datasets

To verify the effectiveness of COLT, we utilize two datasets for multi-tool scenarios: ToolBench and a newly constructed dataset, TOOLLENS.

ToolBench. ToolBench (Qin et al., 2023b) is a benchmark commonly used to evaluate the capability of LLMs in tool usage. For our experiments, we have chosen its I3 subset. After preprocessing, it comprises 23,734 queries and 1,419 tools, with each query linked to 2-4 ground-truth tools.

TOOLLENS. While existing datasets like ToolBench (Qin et al., 2023b) and TOOLE (Huang et al., 2023) provide multi-tool scenarios, they present limitations. TOOLE encompasses merely 497 queries, and ToolBench's dataset construction, which involves providing complete tool descriptions to ChatGPT, results in verbose and semantically direct queries. These do not accurately reflect the brief and often multifaceted nature of real-world user queries. To address these shortcomings, we introduce TOOLLENS, crafted specifically for multi-tool scenarios.

As shown in Figure 3, the creation of TOOLLENS involves a novel five-step methodology: **1) Tool Selection:** Starting with the diverse tool set from ToolBench, we filter out tools not applicable to everyday user queries, such as those for authentication or testing, retaining 464 high-quality, callable tools. **2) Scene Mining:** Utilizing GPT-4, we generate scenes relevant to the detailed descriptions of the selected tools. **3) Query Generation:** We then employ GPT-4 to craft queries based on the provided scene and the parameters required for tool calling. **4) Tool Aggregation:** To enhance the relevance of queries across multiple tools, we reprocess them through GPT-4 to identify categories of potentially applicable tools, which are then aligned with our tool set through dense retrieval and manual verification. **5) Query Rewriting:** Finally, GPT-4 reformulate the queries to include essential parameters, yielding concise yet intentionally multifaceted queries that better mimic real-world user behaviors.

This comprehensive construction pipeline ensures TOOLLENS accurately simulates the dynamics of real-world tool retrieval scenarios. For a detailed description of the dataset construction process, please refer to Appendix A.1. Through the outlined method, we construct the TOOLLENS dataset, featuring 18,770 queries and 464 tools, with each query linked to 1-3 ground-truth tools.

Discusion. Different from prior datasets for tool learning, TOOLLENS uniquely focuses on creating queries that are natural, concise, and intentionally multifaceted to more accurately reflect the complex demands in real-world scenarios. Furthermore, we evaluate the quality of TOOLLENS, finding that it is generally better than both ToolBench and TOOLE, particularly in creating natural and multifaceted queries, as detailed in Appendix A.2.

4 Experiments

In this section, we first describe the experimental setups and then conduct an extensive evaluation and analysis of the proposed COLT.

Methods	ToolLens						ToolBench					
	R@3	R@5	N@3	N@5	C@3	C@5	R@3	R@5	N@3	N@5	C@3	C@5
BM25	21.58	26.88	23.19	26.09	3.89	6.13	29.33	35.88	32.20	35.08	5.52	9.78
ANCE	80.62	94.17	82.35	90.15	54.23	85.83	65.11	76.63	69.27	74.14	34.68	53.64
+COLT (Ours)	**92.15**	**97.78**†	**92.78**	**96.10**	**80.50**	**94.40**	**73.37**	**83.97**	**77.95**	**82.14**	**46.01**	**66.41**
TAS-B	81.26	94.06	82.54	89.94	54.66	85.72	66.04	77.64	70.41	75.34	35.69	55.75
+COLT (Ours)	**91.49**	**96.91**	**92.48**	**95.63**	**79.00**	**92.22**	**74.49**	**84.58**	**79.03**	**82.95**	**48.16**	**68.35**
coCondensor	82.37	94.69	83.90	91.06	56.37	86.73	66.97	79.30	71.20	76.50	37.08	58.66
+COLT (Ours)	**92.65**	**97.78**†	**93.16**	**96.17**	**82.25**	**94.56**†	**75.48**	**84.97**	**80.00**	**83.55**	**49.17**	**68.64**†
Contriever	83.58	95.17	84.98	91.69	59.46	88.65	68.58	80.05	72.86	77.69	39.70	60.89
+COLT (Ours)	**93.64**†	**97.75**	**94.53**†	**96.91**†	**84.55**†	**94.08**	**76.63**†	**85.50**†	**81.21**†	**84.18**†	**52.00**†	**68.47**

Table 1: Performance comparison of different tool retrieval methods on ToolLens and ToolBench datasets. "†" denotes the best results for each column. "+COLT (Ours)" indicates that dense retrieval backbones are equipped with our proposed method. R@K, N@K, and C@K are short for Recall@K, NDCG@K and Comp@K, respectively.

4.1 Experimental Setups

Evaluation Metrics. As discussed in Figure 1(b), traditional retrieval metrics like Recall and NDCG do not adequately fulfill the requirements of retrieval completeness that are crucial for effective tool retrieval. To further tailor our assessment to the specific challenges of tool retrieval tasks, we also introduce a new metric, Comp@K. This metric is designed to measure whether the top-K retrieved tools form a complete set with respect to the ground-truth set:

$$\text{COMP}@K = \frac{1}{|\mathcal{Q}|}\sum_{q=1}^{|\mathcal{Q}|}\mathbb{I}(\Phi_q \subseteq \Psi_q^K),$$

where Φ_q denotes the set of ground-truth tools for query q, Ψ_q^K represents the top-K tools retrieved for query q, and $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if the retrieval results include all ground-truth tools within the top-K results for query q, and 0 otherwise.

Baselines. COLT is benchmarked against several established methods, including the lexical retrieval model BM25 (Robertson et al., 2009) and four state-of-the-art PLM-based dense retrieval models: ANCE (Xiong et al., 2020), TAS-B (Hofstätter et al., 2021), coCondensor (Gao and Callan, 2021), and Contriever (Izacard et al., 2021). For more details, please refer to Appendix D.1.

4.2 Experimental Results

Retrieval Performance. Table 1 presents the overall results of different tool retrieval methods on ToolLens and ToolBench. From the results, we have the following observations and conclusions:

	Evaluation Aspects			
	Coherence	Relevance	Comprehensiveness	Overall
BM25	848	845	860	780
ANCE	934	936	946	1016
TAS-B	995	991	988	1028
coCondensor	1031	1036	1041	1035
Contriever	1076	1082	1044	1046
COLT (Ours)	1116	1110	1121	1096

Table 2: Elo ratings for different models w.r.t. "Coherence", "Relevance", "Comprehensiveness" and "Overall" evaluated by GPT-4.

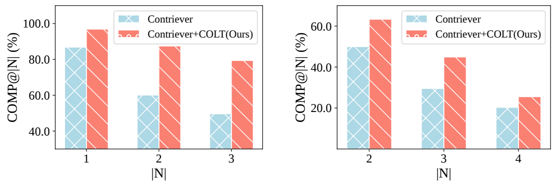
The BM25 model significantly lags behind PLM-based dense retrieval methods, highlighting the superior performance of the latter in leveraging contextual information for tool retrieval. Despite this advantage, PLM-based methods fall short in the Comp metric, designed specifically for evaluating completeness in tool retrieval scenarios. This suggests that while effective for general retrieval tasks, PLM-based methods may not fully meet the unique demands of tool retrieval.

All base models equipped with COLT exhibit significant performance gains across all metrics on both datasets, particularly in the Comp@3 metric. These improvements demonstrate the effectiveness of COLT, which can be attributed to that COLT adopts a two-stage learning framework with semantic learning followed by collaborative learning. In this way, COLT can capture the intricate collaborative relationships between tools, resulting in effectively retrieving a complete tool set.

Downstream Tool Learning Performance. To verify that improvements of COLT in tool retrieval genuinely enhance the downstream real-world tool learning applications, we further conduct a validation study using GPT-4. Specifically, we randomly select 100 queries from the ToolLens test set

Methods	ToolLens R@|N|	ToolLens C@|N|	ToolBench R@|N|	ToolBench C@|N|
Contriever+COLT (Ours)	92.76	82.95	75.40	49.81
w/o semantic learning	65.21	30.90	53.33	19.63
w/o collaborative learning	80.60	54.44	68.20	36.91
w/o contrastive learning	84.58	60.52	69.46	39.02

Table 3: Ablation study of the proposed COLT.



(a) ToolLens (b) ToolBench

Figure 4: Performance comparison regarding different sizes of ground-truth tool sets.

and use various retrieval models to return the top-3 tools for each query. We then utilize GPT-4 as an evaluator, examining the responses generated with different retrieved tools across four dimensions: Coherence, Relevance, Comprehensiveness, and Overall. Finally, we employ a pairwise evaluation method and use Elo ratings to demonstrate the performance, with details provided in Appendix C.

The results in Table 2 show that superior tool retrieval models can significantly improve downstream tool learning performance. Moreover, responses generated with the retrieved tools from COLT notably outperform those from other methods, achieving the highest Elo ratings in all four assessed dimensions. These results highlight the pivotal role of effective tool retrieval in enhancing the performance of downstream applications and further confirm the superiority of COLT.

4.3 Further Analysis

Next, we delve into investigating the effectiveness of COLT with Contriever as the base model since it achieves the best performance when equipped with COLT in Table 1. The results with other dense retrieval models as backbones demonstrate similar trends and are provided in Appendix D.2. Recall@|N| and Comp@|N| are adopted as evaluation metrics, with |N| representing the count of ground-truth tools suitable to each query.

Ablation Study. We conduct ablation studies to assess the impact of various components within our COLT. The results presented in Table 3, highlight the significance of each element:

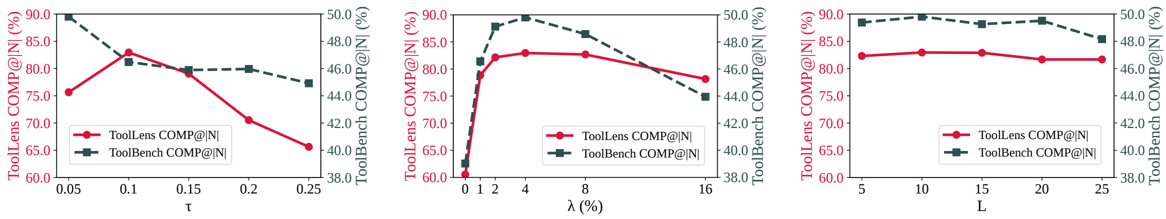
***w/o* semantic learning** denotes an off-the-shelf PLM is directly employed to get the initial representation for the following collaborative learning stage without semantic learning on the given dataset in § 2.3. The absence of semantic learning significantly diminishes the performance, confirming its essential role in aligning the representations of tools and queries as the basic of the following collaborative learning.

***w/o* collaborative learning** is a variant that the collaborative learning state is omitted (*i.e.,* only semantic learning). The significant decline in performance in this variant further supports the effectiveness of COLT in capturing the high-order relationships between tools through graph collaborative learning, thereby achieving a comprehensive tool retrieval.

***w/o* contrastive learning** refers to a variant that optimizes without the contrastive learning loss defined in Eq. (8) and (9); This omission also leads to a noticeable performance drop, emphasizing the benefit of introducing contrastive learning to achieve better representation for queries and tools from a dual-view learning framework.

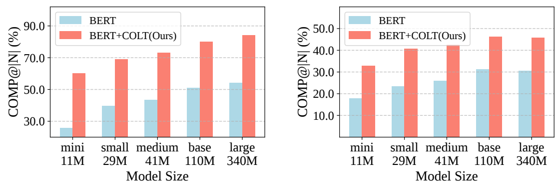
Performance w.r.t. Different Tool Sizes. The ToolLens dataset encompasses queries that require 1-3 tools, while ToolBench includes queries needing 2-4 tools. To assess how well COLT adapts to queries with diverse tool requirements, we divide each dataset into three subsets according to the number of tools required by each query and conduct a focused analysis on these subsets. As shown in Figure 4, there is a discernible decline in performance as the number of ground-truth tools increases, reflecting the escalating difficulty of achieving complete retrieval. However, COLT demonstrates consistent performance improvement across all subsets. This improvement is especially significant in the most challenging cases, where queries require up to three or four tools. These results consistently highlight the robustness of COLT and its potential to meet the complex demands of tool retrieval tasks across various scenarios.

Performance w.r.t. Model Size of PLM. To verify the adaptability and effectiveness of COLT across varying sizes of PLMs, we explore its integration with a range of BERT models, from BERT-mini to BERT-large. This analysis aims to determine whether COLT could generally enhance tool retrieval performance across different model sizes. Figure 6 presents the results, illustrating a clear trend: while the performance of the base model nat-



(a) Temperature τ. (b) Loss weight λ. (c) List length L.

Figure 5: Sensitivity analysis of COLT performance to hyper-parameters. (a) shows the dependency of model performance on temperatureτ. (b) illustrates the influence of loss weight λ. (c) examines the effect of list length L.



(a) ToolLens (b) ToolBench

Figure 6: Comparison of different model sizes of PLM.

urally improves with larger PLM sizes, the equipping of COLT consistently boosts performance across all sizes. Remarkably, even BERT-mini equipped with COLT, significantly outperforms a much larger BERT-large model (30x larger) operating without our COLT. These results underscore the generalization and robustness of COLT, proving its potential to significantly improve tool retrieval performance for PLMs of any scale.

Hyper-parameter Analysis. Figure 5 illustrates the sensitivity of COLT to the temperature parameter τ and the loss weight λ, but shows relative insensitivity to variations in the sampled list length L. The influence of τ varies across two datasets, suggesting that its impact is dependent on the specific data distribution. Conversely, the pattern observed for λ across both datasets is consistent, marked by an initial performance improvement that eventually plateaus, underscoring the importance of carefully selecting λ to maximize the effectiveness of COLT.

5 Related Work

Tool Learning. Recent studies highlight the potential of LLMs to utilize tools in addressing complex problems (Qin et al., 2023a; Mialon et al., 2023). Existing tool learning approaches can be categorized into two types: tuning-free and tuning-based methods (Gao et al., 2024). Tuning-free methods rely on black-box LLMs such as ChatGPT, employing in-context learning through demonstration examples for tool invocation (Shi et al., 2023). Conversely, tuning-based methods fine-tune LLMs on specific datasets to master tool usage and amalgamate diverse reasoning strategies such as ReAct (Yao et al., 2022), DFSDT (Qin et al., 2023b), and self-consistency (Wang et al., 2022) to enhance the inferential abilities of LLMs.

Tool Retrieval. Tool retrieval aims at finding top-K most suitable tools for a given query from a vast tool set. State-of-the-art retrieval methods can be categorized into two types: term-based and semantic-based. Term-based methods, such as TF-IDF (Sparck Jones, 1972) and BM25 (Robertson et al., 2009), prioritize term matching via sparse representations. Conversely, semantic-based methods utilize neural networks to learn the semantic relationship between queries and tool descriptions (Xiong et al., 2020; Hofstätter et al., 2021), and then calculate the semantic similarity using methods such as cosine similarity. Despite the advancements, existing tool retrieval methods overlook the importance of the collaborative potential among multiple tools, thereby falling short of meeting the completeness criterion for tool retrieval. Our work tries to deal with these issues through the incorporation of graph collaborative learning.

6 Conclusion

This study introduces COLT, a novel two-stage approach designed to enhance the completeness of tool retrieval tasks. By incorporating graph collaborative learning and cross-view contrastive learning, COLT captures the collaborative relationships among tools. Extensive experimental results and analysis demonstrate the effectiveness of COLT, especially in handling multifaceted queries with multiple tool requirements. Furthermore, we release a new dataset ToolLens and introduce a novel evaluation metric Comp, both of which are valuable resources for tool retrieval.

Limitations

In our study, we showcase the efficacy of our innovative tool retrieval method, COLT. However, COLT is developed without the incorporation of LLMs, which possess a greater number of parameters and wield stronger capabilities. Considering that tool retrieval ultimately serves LLMs, integrating them into the retrieval process could potentially enhance the performance of tool retrieval. Moreover, certain tool combinations involve sequential calls, where the output of one tool might serve as input for another. Unfortunately, COLT does not account for such interactions. Future work could explore ways to incorporate LLMs and dependencies among tools into the method, thereby improving the performance of tool retrieval.

Ethics Statement

In this work, we present a new dataset towards tool retrieval, TOOLLENS, which is created through the utilization of GPT-4 and subsequent human manual quality checks. We believe that our dataset will help advance the field of tool retrieval and tool learning by providing high-quality tool set and quries for researchers.

The tools in our TOOLLENS are sourced from publicly available sources, and queries are generated through GPT-4. So it does not encompass issues related to user privacy. Moreover, the use of human manual quality check ensures a higher level of quality in TOOLLENS, but it also raises ethical considerations. Given that the evaluation of quality for queries relies on common sense, which can differ significantly among individuals from various backgrounds. We acknowledge the potential for human annotation to harbor errors or biases.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*, RecSys '23. ACM.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms.

Luyu Gao and Jamie Callan. 2021. Unsupervised corpus aware language model pre-training for dense passage retrieval. *arXiv preprint arXiv:2108.05540*.

Shen Gao, Zhengliang Shi, Minghang Zhu, Bowen Fang, Xin Xin, Pengjie Ren, Zhumin Chen, Jun Ma, and Zhaochun Ren. 2024. Confucius: Iterative tool learning from introspection feedback by easy-to-difficult curriculum. In *AAAI*.

Jiafeng Guo, Yinqiong Cai, Yixing Fan, Fei Sun, Ruqing Zhang, and Xueqi Cheng. 2022. Semantic models for the first-stage retrieval: A comprehensive review. *ACM Transactions on Information Systems (TOIS)*, 40(4):1–42.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–122.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems.

Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. 2023. Metatool benchmark: Deciding whether to use tools and which to use. *arXiv preprint arXiv: 2310.03128*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. API-bank: A comprehensive benchmark for tool-augmented LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3102–3116, Singapore. Association for Computational Linguistics.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2023. Holistic evaluation of language models.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023a. Lost in the middle: How language models use long contexts.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. G-eval: Nlg evaluation using gpt-4 with better human alignment.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. 2023. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*.

Aaron Parisi, Yao Zhao, and Noah Fiedel. 2022. Talm: Tool augmented language models. *arXiv preprint arXiv:2205.12255*.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, et al. 2023a. Tool learning with foundation models. *arXiv preprint arXiv:2304.08354*.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. 2023b. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Andrea Sottana, Bin Liang, Kai Zou, and Zheng Yuan. 2023. Evaluation metrics in the era of GPT-4: Reliably evaluating large language models on sequence to sequence tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8776–8788, Singapore. Association for Computational Linguistics.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Weiwei Sun, Zheng Chen, Xinyu Ma, Lingyong Yan, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Instruction distillation makes large language models efficient zero-shot rankers.

Raphael Tang, Xinyu Zhang, Xueguang Ma, Jimmy Lin, and Ferhan Ture. 2023. Found in the middle: Permutation self-consistency improves listwise ranking in large language models.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. 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)*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Minghao Wu and Alham Fikri Aji. 2023. Style over substance: Evaluation biases for large language models.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Junjie Ye, Guanyu Li, Songyang Gao, Caishuang Huang, Yilong Wu, Sixian Li, Xiaoran Fan, Shihan Dou, Qi Zhang, Tao Gui, et al. 2024. Tooleyes: Fine-grained evaluation for tool learning capabilities of large language models in real-world scenarios. *arXiv preprint arXiv:2401.00741*.

Siyu Yuan, Kaitao Song, Jiangjie Chen, Xu Tan, Yongliang Shen, Ren Kan, Dongsheng Li, and Deqing Yang. 2024. Easytool: Enhancing llm-based agents with concise tool instruction. *arXiv preprint arXiv:2401.06201*.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2023. Dense text retrieval based on pretrained language models: A survey. *ACM Trans. Inf. Syst.*

Appendix

A TOOLLENS DATASET DETAILS

In this section, we introduce the TOOLLENS dataset details, featuring 18,770 user queries and 464 tools.

A.1 Details Of Dataset Construction

We propose a novel five-step methodology to construct our dataset:

• **Tool Selection:** In order to construct a high-quality tool dataset, we initially conduct rigorous filtering of the original tool collection from ToolBench, focusing on tools that are both existent and directly callable. To better tailor our dataset to the practical query requirements of real-world users, we exclude tools requiring authentication, testing, and ID mapping, which are not applicable to everyday user queries;

• **Scene Mining:** As shown in Table 8 (I), we design an instruction and require GPT-4 to generate potential scenes relevant to the detailed descriptions of the selected tools;

• **Query Generation:** As shown in Table 8 (II), we then design an another instruction to employ GPT-4 to craft user queries based on the provided scene and only the parameters of the tool (not the whole tool description);

• **Tool Aggregation:** The queries generated in aforementioned way are only relevant to a single tool. To enhance the relevance of the query across multiple tools, as shown in Table 8 (III), we utilize GPT-4 to generate the categories of tools potentially capable of resolving the query. To align the generated categories of tools with our existing tool set, the query's originating tool is first matched to one of these categories. we then utilize dense retrieval to retrieve the most relevant tools within our tool set corresponding to the remaining categories, which are then designated as the ground-truth tool for the query.

• **Query Rewriting:** As shown in Table 8 (IV), to ensure the query comprehensively includes all necessary parameters for invoking tools, we utilizing GPT-4 to revise the query to encompass all the essential parameters by providing it with both the initial query and a list of necessary parameters, thereby producing the final query.

A.2 Quality Verification

To assess the quality of TOOLLENS, following previous works (Gao et al., 2024; Liu et al., 2023b; Sottana et al., 2023), we employ GPT-4 as evaluator and human evaluation where three well-educated doctor students are invited to evaluate 50 randomly sampled cases from TOOLLENS, ToolBench and TOOLE in the following two aspects:(1) Natural-query: whether the query is natural. (2) Multi-faceted intentions: whether the user intent is multi-faceted. The results are illustrated in Table 4. In most cases, TOOLLENS outperforms ToolBench and TOOLE. Furthermore, using GPT-4 as the evaluator shows a high degree of consistency with human evaluation trends, which underscores the validity of employing GPT-4 as an evaluator.

Evaluator	TOOLLENS vs. ToolBench			TOOLLENS vs. TOOLE		
	Whether the query is natural?					
GPT-4	TOOLLENS 68%	ToolBench 14%	Equal 18%	TOOLLENS 44%	TOOLE 36%	Equal 20%
Human	TOOLLENS 64%	ToolBench 10%	Equal 26%	TOOLLENS 54%	TOOLE 24%	Equal 22%
	Whether the user intent is multifaceted?					
GPT-4	TOOLLENS 62%	ToolBench 14%	Equal 24%	TOOLLENS 50%	TOOLE 24%	Equal 26%
Human	TOOLLENS 60%	ToolBench 12%	Equal 28%	TOOLLENS 58%	TOOLE 18%	Equal 24%

Table 4: Quality verification of TOOLLENS.

A.3 Case Study

Table 5 displays the comparison of three typical instances of the user queries in TOOLLENS and ToolBench datasets. The queries within TOOLLENS, crafted through our innovative five-step methodology, exhibit notable distinctions from those in the ToolBench dataset, underscoring the superior-

Scene	TOOLLENS	ToolBench
Cooking	I'm planning a meal using the ingredient beef.	I'm organizing a dinner party for my friends and I need some recipe suggestions. Please provide me with a variety of chicken recipes and their nutritional information. Also, I would appreciate some cocktail recommendations to complement the meal.
Travel	I want to travel to Paris.	I'm planning a family vacation to London in August. Can you help me find the best hotels in London for a family of four? Also, provide me with the distance between Birmingham, Alabama, and Sacramento, California. Lastly, recommend some fun activities for kids in London.
Investment	I am currently tracking my cryptocurrency investments.	I'm a teacher and I want to plan an engaging lesson on current events for my students. Can you provide me with specific articles related to bitcoin from different news sources? Additionally, I'd like to gather information about crypto news and the latest trends in the market. Finally, could you recommend any climate change news from reliable sources to discuss the impact on the economy?

Table 5: The comparison of some typical queries in the TOOLLENS and ToolBench datasets.

Algorithm 1 The Learning Algorithm of COLT

Input: PLM, semantic learning training epoch E,Query-scene bipartite graph, query-tool bipartite graph, scene-tool bipartite graph, learning rate lr, weight decay, layer number I, contrastive loss weight λ, temperature coefficient τ, list length L;
Parameter: Learnable parameters θ;
Output: COLT Model;

Semantic Learning:
for $e = 1$ **to** E **do**
  Calculate the MultipleNegativesRankingLoss;
  Update parameter of PLM using AdaW;
end for
Collaborative Learning:
Calculate inital $\mathbf{e}_q^{S(0)}$, $\mathbf{e}_s^{S(0)}$, $\mathbf{e}_q^{T(0)}$ and $\mathbf{e}_t^{T(0)}$ using the embeddings obtained from the first-stage semantic learning and Eq. (2);
while COLT Not Convergence **do**
  for $i = 1$ **to** I **do**
    Conduct message propagation using Eq. (1) and Eq. (4);
  end for
  Calculate final $\mathbf{e}_q^{S}$, $\mathbf{e}_s^{S}$, $\mathbf{e}_q^{T}$, $\mathbf{e}_s^{T}$ and $\mathbf{e}_t^{T}$ using Eq. (3), Eq. (5) and Eq. (6);
  Calculate two-view contrastive loss $\mathcal{L}_Q^C$ and $\mathcal{L}_S^C$ using Eq. (8) and Eq. (9);
  Calculate multi-label loss $\mathcal{L}_{\text{list}}$ using Eq. (12);
  Calculate total loss $\mathcal{L}$ using Eq. (13);
  Update model parameter using Adam;
end while
return θ

ity of our dataset. As can be observed, the queries within TOOLLENS are concise, appear more natural and more multifaceted. For example, in the travel scene, the query from TOOLLENS may simply express a desire to visit Paris, making them more natural and more multifaceted. Conversely, the ToolBench dataset contains queries that explicitly seek information about hotels, distance and tourist attractions.

B Algorithm

The learning algorithm of COLT is shown in Algorithm 1.

C Evaluation Details

C.1 Pairwise Comparison

Pairwise ranking is a powerful method used to improve search results and the performance of recommendations (Dai et al., 2023; Sun et al., 2023; Liang et al., 2023). In these systems, pairwise ranking involves comparing pairs of items, such as web pages, products, or documents, to identify which one is more relevant or preferable in response to a user query or profile.

To assess response quality, we employ a pairwise comparison approach. In this method, the user query and a pair of responses are utilized as prompts to guide GPT-4 in determining the superior response. Additionally, we also consider that LLMs may respond differently to the order in which text is presented in the prompt (Lu et al., 2022; Tang et al., 2023; Hou et al., 2024; Liu et al., 2023a). To mitigate potential biases associated with this, we execute each comparison twice, reversing the response order for the second evaluation which ensures a more reliable assessment.

C.2 Elo Ratings

Utilizing GPT-4 for pairwise comparisons, we establish a tournament-style competition where models compete against each other on Elo ratings system that are widely employed in chess and other two-player games to measure the relative skill levels of the players (Dettmers et al., 2023; Wu and Aji, 2023). Each player is assigned an Elo score. Given two players a and b with their Elo ratings R_a and R_b, the expected score E_a and E_b for these two players are:

$$E_a = \frac{1}{1 + 10^{\frac{R_b - R_a}{400}}}, E_b = \frac{1}{1 + 10^{\frac{R_a - R_b}{400}}}.$$

For instance, a player with an Elo of 1100 competing against a player with an Elo of 1000 has an expected win rate of approximately 65%, When both players have the same Elo score, their expected win rate against each other is 50%. The outcome of matches between players leads to adjustments in their Elo scores, the updated Elo ratings $R_a^{'}$ and $R_b^{'}$ are:

$$R_a^{'} = R_a + K(S_a - E_a), R_b^{'} = R_b + K(S_b - E_b),$$

where S_a and S_b are the actual score of players a and b, K is the K-factor that determines the maximum amount of points a player's rating can change from a single game outcome.

Following previous works (Chiang et al., 2023), we start with a score of 1000 and set $K = 32$. Additionally, in order to reduce the influence of match sequences on Elo score computations, we repeatedly conduct these calculations 10000 times using different random seeds, ensuring the control of ordering effects.

C.3 Case Study

Table 9 displays the comparison of responses generated by GPT-4 that integrates information from

Dataset	# Query	# Query In Training	# Query In Testing
ToolLens	18,770	16,893	1,877
ToolBench	23,734	21,361	2,373

Table 6: Statistics of the experimental datasets.

multiple tools retrieved through different tool retrievers in response to user queries. As can be observed, due to the more comprehensive and complete tools retrieved by COLT, the responses generated by GPT-4 based on the tools offered by COLT are more comprehensive and effective.

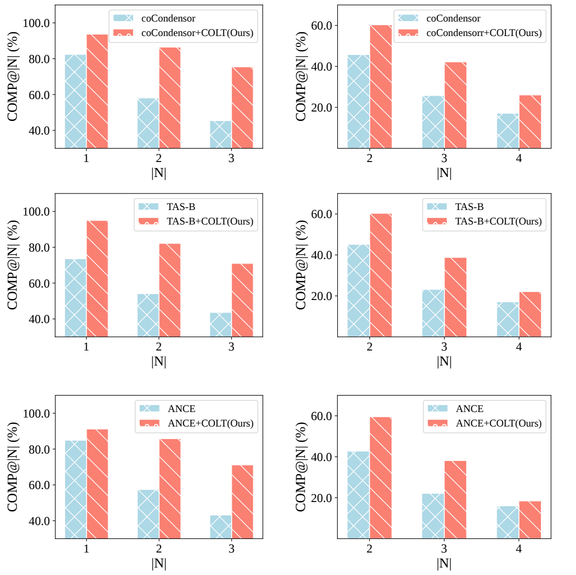
D More Experiments

D.1 More Details Of Experimental Setups

Datasets. We randomly selected 10% of the entire dataset to serve as the test data. The statistics of both datasets after preprocessing are summarized in Table 6.

Baselines. BM25 (Robertson et al., 2009) is a classical lexical retrieval model that employs an inverted index for identifying suitable tools based on the exact term matching. ANCE (Xiong et al., 2020) is a dense retrieval model that employs a dual-encoder architecture to globally select hard negatives across the entire corpus via an asynchronously updated ANN index for training. TAS-B (Hofstätter et al., 2021) is a bi-encoder trained with a balanced margin sampling technique that samples queries out of a cluster per batch, ensuring efficiency in the sampling process. coCondenser (Gao and Callan, 2021) incorporates a query-agnostic contrastive loss based on the retrieval corpus that clusters text segments from the same document while distinguishing unrelated segments for enhanced retrieval performance. Contriever (Izacard et al., 2021) employs inverse cloze task and cropping for generating positive pairs and momentum contrastive learning for training dense retrievers, achieving state-of-the-art zero-shot retrieval performance.

Implementation Details. We utilize the BEIR (Thakur et al., 2021) framework to implement the dense retrieval baselines, set the training epochs to 5 with the learning rate of $2e$–5, weight decay of 0.01, and using the AdamW optimizer. As our approach is model-agnostic, we directly apply dense retrieval for the semantic learning stage, and in the collaborative learning stage, we set the batch size as 2048 and carefully tune the hyper-parameters learning rate, weight decay, layer number I, contrastive loss weight λ, temperature coeficient τ, list length L among $\{1e–3, 5e–3, 1e–4, 5e–4, 1e–5\}$, $\{1e–5, 1e–6, 1e–7\}$, $\{1, 2, 3\}$, $\{0.02, 0.04, 0.1\}$, $\{0.05, 0.1, 0.15, 0.2, 0.25\}$, $\{5, 10, 15, 20, 25\}$. All the experiments are conducted on NVIDIA RTX A6000 48G GPUs using Ubuntu 18.04.1 SMP.



(a) ToolLens (b) ToolBench

Figure 7: Performance comparison at different sizes of ground-truth tool sets of COLT using coCodensor, TAS-B, ANCE as the backbone.

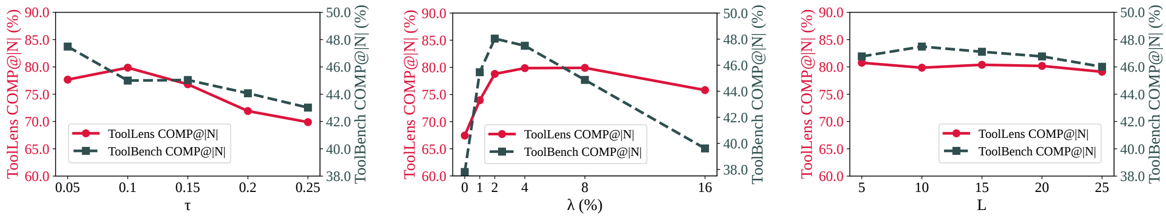
D.2 More Results and Analysis

We also conduct analysis experiments using other PLM-based dense retrieval models as the backbone.

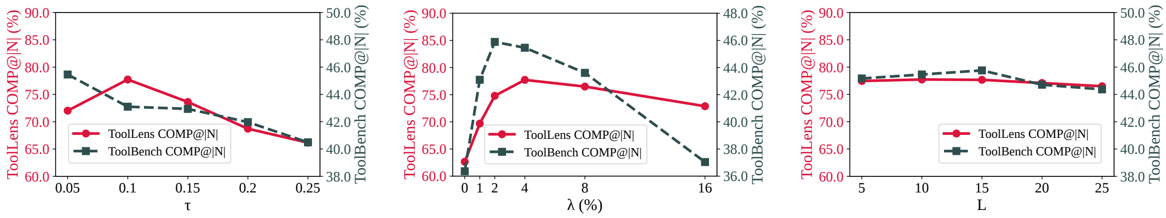
Performance w.r.t. Different Tool Sizes. As shown in Figure 7, using coCondensor, TAS-B, ANCE as the backbone exhibit similar trends to Contriever. our method demonstrates consistent efficacy across all subsets and backbones, achieving significant improvements in retrieval completeness under various testing conditions.

Hyper-parameter Analysis. As shown in Figure 8, using coCondensor, TAS-B, ANCE as the backbone reveals similar trends to those oberved with the Contriever backbone; Specifically, there is a noticeable sensitivity to temperature τ and lambda λ, while the response to list length L is insensitive.

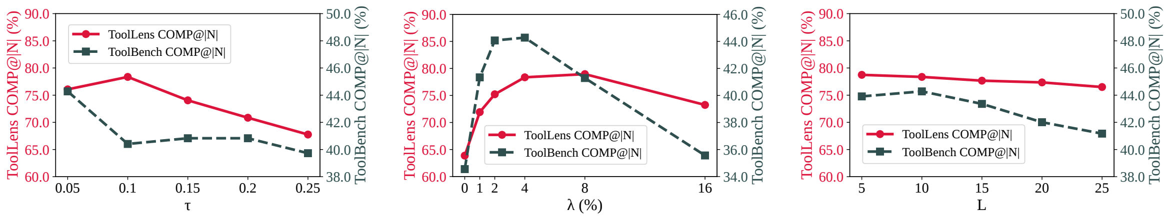
Ablation Study. As shown in Table 7, we conduct ablation experiments to assess the effectiveness of



(a) coCondensor.



(b) TAS-B.



(c) ANCE.

Figure 8: Sensitivity analysis of COLT performance to hyper-parameters. (a) shows using coCondensor as the backbone. (b) shows using TAS-B as the backbone. (c) shows using ANCE as the backbone.

Methods	TOOLLENS		ToolBench	
	R@|N|	C@|N|	R@|N|	C@|N|
coCondensor+COLT (Ours)	91.49	79.86	74.00	47.49
w/o semantic learning	30.38	5.54	25.07	2.27
w/o contrastive learning	86.78	67.07	68.92	37.80
w/o collaborative learning	78.83	50.61	64.38	33.08
TAS-B+COLT (Ours)	90.29	77.73	72.84	45.46
w/o semantic learning	38.49	9.16	32.16	5.47
w/o contrastive learning	84.86	62.65	67.66	36.36
w/o collaborative learning	76.86	47.83	63.61	31.73
ANCE+COLT (Ours)	91.08	78.36	72.22	44.28
w/o semantic learning	36.49	6.84	21.92	1.60
w/o contrastive learning	85.63	63.87	66.57	34.55
w/o collaborative learning	77.36	49.01	62.39	30.12

Table 7: Ablation study of the proposed COLT using coCodensor, TAS-B, ANCE as the backbone.

various design component within COLT using coCondensor, TAS-B, ANCE as the backbone. The results reveal that each component enhances the retrieval performance of COLT, mirroring trends observed when using Contriever as the backbone. Notably, the omission of semantic learning elements markedly reduces performance across coCondensor, TAS-B, and ANCE more so than with Contriever. This highlights Contriever's superior ability in zero-shot learning scenarios compared to the other models, underscoring the importance of semantic learning in initial retrieval stages. Additionally, our analysis indicates that contrastive learning is particularly vital for Contriever, as its absence results in performance lagging behind the other models. This underscores the pivotal role of contrastive learning in refining retrieval efficiency.

E Complete Response

The complete responses illustrated in Figure 1(b) are detailed in Table 10. We can find that the absence of any essential tool from the ground-truth tool set compromises the ability to fully address user queries. Moreover, the transition from having two missing tools to two incorrect ones results in a marked decline in the quality of responses generated by GPT-4. This observation demonstrates that providing redundant tools does not necessarily facilitate problem-solving.. Such shortcomings impede successful query resolution, emphasizing the necessity of both a complete and accurate tools for achieving optimal performance.

I: Scene Mining

/ I: Task prompt */*
Your task is to generate potential scenes where a specific tool function might be used. Below is the description of the tool function. Please provide a scene where this tool function could be utilized. Just give the scene, no explanation needed. Here is an example:
/ Example */*
The given tool description is:
{category_name:Food, tool_name:Worldwide Recipes, api_name: Suggestions, api_description: Get Suggestions, required_parameters:[{"name": "q", "type": "STRING", "description": "", "default": "chicken" }], optional_parameters: [], method: GET, template_response: {"success": "bool", "message": "str", "results": {"ingredients": ["list of str with length 413"], "searches": ["list of str with length 10"]}}}
Your response would be: cooking.
/ Generation of Scene */*
The API description is: ...
Your response is:

II: Query Generation (Single-Tool)

/ Task prompt */*
Your task is to create a potentially multifaceted user instruction based on a given scene. The instruction should be presented as a declarative sentence. It should include both required and optional parameters, for which you need to generate random information. For required parameters like IP address, location, coordinates, etc., provide specific details. For example, instead of simply stating "an address", provide the exact road and district names. Here is an example:
/ Example */*
The given scene is Going out fishing
The given required and optional parameters are as follows:
required_parameters: [{"name": "longitude","type": "STRING","description": "","default": "-79.99"},{"name": "date", "type": "STRING","description": "","default": "2022-01-01"},{"name":"latitude","type": "STRING","description": "", "default": "39.53"}], "optional_parameters": []
Your response would be: I'm planning a fishing trip at the coordinates 51.5074 N, -0.1278 W for the date 2020-06-15.
/ Generation of Query */*
The scenario for this task is: ...
The required and optional parameters are as follows: ...
Your response is:

III: Tool Aggregation (Category Generation)

/ Task prompt */*
You will be provided with a user query, and your task is to provide three types of API functions that can better answer this query.
/ Generation of Categories */*
The user query is: ...
Your response is:

IV: Query Rewritting (Multi-Tool)

/ Task prompt */*
Your task is to rewrite the user query based on the given user query and the required parameters. The original user query may not include all the required parameters, and you need to change it to a user query that includes all the required parameters, for which you need to generate random information. For required parameters like IP address, location, coordinates, etc., provide specific details. For example, instead of simply stating 'an address', provide the exact road and district names. Try to rewrite the user query as if it were a human inquiry, without dividing it into three separate parts.
/ Generation of Query */*
The user query is: ...
The required parameters are as follows: ...
Your response is:

Table 8: The prompt for GPT-4 for scene generation, query generation (single-tool) and query rewrite (multi-tool).

Query *Input query:* I'm planning a picnic outing in Helsinki.
BM25: That sounds wonderful! Helsinki has a number of beautiful parks that would be perfect for a picnic. Just to name a few, there are the Esplanade Park, Kaisaniemi Park, and Kaivopuisto Park. These parks have areas with picnic tables and open fields where you could spread out a blanket. Enjoy your picnic!
ANCE: Planning a picnic in Helsinki can be a wonderful experience, as the city offers many beautiful parks and natural settings. Here are some steps and tips to help you plan your outing:1. Choose a Date and Check the Weather: Helsinki's weather can be quite variable, so check the forecast to choose a date with pleasant weather. During summer, the days are warm and long, making it an ideal time for picnics. 2. Select a Location: Some popular picnic spots in Helsinki include: Suomenlinna: A sea fortress and UNESCO World Heritage site offering green areas with sea views. Kaivopuisto: A park with wide green spaces, views of the sea, and nice spots for picnics. Enjoy your picnic in Helsinki, and remember to soak in the beauty of the Finnish capital!
TAS-B: That sounds like a great plan! Here are some wonderful picnic spots in Helsinki: 1. Kaivopuisto Park: This park is one of the oldest and best known in Helsinki, providing stunning views over the sea. 2. Esplanade Park: Located in the city center with lots of cafes and shops nearby. Concerts and events often take place here during summer. 3. Suomenlinna: A picnic on the fortress island of Suomenlinna is a unique experience. After a picnic, you can tour the historic fortifications and visit the museums on the island. 4. Seurasaari Island: A great place for an outdoor picnic. You can also tour the open-air museum and feed the squirrels on this island. 5. Hietaniemi Beach: If you want a picnic by the beach, this is a great spot. It's a great place for swimming, too. Enjoy your picnic!
coCondensor: When planning a picnic outing in Helsinki, consider the following tips: 1. Check the Weather: Helsinki has a northern continental or subarctic climate, so always check the weather forecast ahead of time. Summers can be pleasant for picnics, usually from late May through August, with temperatures ranging from 15°C to 25°C (59°F to 77°F). Remember that weather can be unpredictable, so it's good to have a backup plan in case of rain. 2. Select a Location: Some popular parks and coastal areas in Helsinki perfect for picnics are: Kaivopuisto: Located by the sea, offering lovely views and open green spaces. Suomenlinna: A sea fortress that's been turned into a park, accessible by ferry, providing a historic setting for a picnic. Töölönlahti: A bay area surrounded by a park in the center of the city, ideal for a relaxed outdoor meal. Enjoy your picnic outing, and remember to be prepared for any changes in weather, protect the environment, and savor the local flavors!
Contriever: Planning a picnic outing in Helsinki is a great idea, especially during the warmer months. Here's a helpful guide to assist you in organizing a lovely picnic: 1. Check the Weather: Make sure you monitor the weather forecast for Helsinki, so you pick a day with pleasant weather. The best months for a picnic are typically May through September when the temperature is moderate and the risk of rain is lower. 2. Select a Location: Kaivopuisto Park: Offers beautiful sea views and lots of green spaces for a picnic. Suomenlinna: A sea fortress island that provides historical sights as well as picnic areas. Töölönlahti Bay: Surrounded by a park that features walking paths and green areas perfect for picnicking. Esplanadi Park: Located in the heart of Helsinki, this park is perfect for people-watching and enjoying urban nature. Seurasaari Island: Gives you a taste of Finnish nature and traditional architecture, with plentiful picnic spots. 3. Prepare Your Picnic Basket: Fill it with snacks and food items that do not require immediate refrigeration. Don't forget to bring drinks, and if you fancy something local, pick up some Finnish rye bread and cold cuts for sandwiches, or grab Karelian pastries from a local bakery. Enjoy your picnic!
COLT: When planning a picnic outing in Helsinki, here are some steps and tips to consider: 1. Choose a Location: Helsinki offers many stunning spots for picnic outings. Some of the popular locations include: Suomenlinna Island: A UNESCO World Heritage site accessible by ferry, offering beautiful views and green spaces. Kaivopuisto: A seaside park with large grassy areas, perfect for a picnic and views of the sea. Central Park (Keskuspuisto): A large park area offering a natural setting for a quiet and relaxing picnic. Töölönlahti Bay: A bay area surrounded by parks where you can enjoy a waterside picnic. 2. Check the Weather: The weather in Helsinki can be unpredictable, so before you head out, check the current weather conditions and forecast. Make sure you bring appropriate clothing for the weather, and consider bringing an umbrella or waterproof blanket if there's a chance of rain. 3. Pack Your Picnic: Prepare a basket or cooler with your favorite picnic foods. Don't forget to pack: Food: Sandwiches, fruits, salads, snacks, etc. Beverages: Water, juice, or soft drinks; and if you desire, a bottle of wine or champagne (but make sure to check local laws regarding alcohol consumption in public spaces). Utensils: Plates, cutlery, napkins, and cups if needed. Extras: Condiments, a cutting board, a knife, and a bottle opener. 4. Bring Comfort Items: For a comfortable picnic, consider packing: A large blanket to sit on. Cushions or foldable chairs for extra comfort. Sunscreen and sunglasses on a sunny day. 5. Entertainment and Games: Change things up from just eating, to also having some fun. Bring along: A frisbee, soccer ball, or other sports equipment. A kite if it's a windy day. Board games, playing cards, or books for a relaxed time. 6. Respect the Environment: When you're ready to leave, make sure to clean up after yourself. Bring trash bags and recycle any waste if possible. 7. Local Amenities: Check if there are nearby restrooms and if pets are allowed if you plan to bring one. 8. Transportation: Helsinki has an excellent public transportation system including buses, trams, metro, and ferries. If you plan to visit somewhere like Suomenlinna, check the ferry schedule ahead of time. For locations within the city, you can use HSL's journey planner to find the best route. Now you're all set for your picnic in Helsinki! Enjoy the beautiful scenery and the tasty food with good company.

Table 9: The comparison of responses generated by Large Language Model integrating information from multiple tools retrieved via different tool retrievers in response to user query.

Query *Input query:* I would like to know the value of 5 ounces of gold plus 1 million AMZN stocks in CNY.
With Complete Tools: First, let's calculate the value of 5 ounces of gold in USD: 5 ounces * 1919.7605807522993 USD/ounce = 9598.802903761497 USD. Then, let's calculate the value of 1 million AMZN stocks in USD: 1,000,000 stocks * 130 USD/stock = 130,000,000 USD. Now, let's convert these values to YNC using the exchange rate: 9598.802903761497 USD * 7.24 YNC/USD = 69,495.48654289563 YNC for the gold. 130,000,000 USD * 7.24 YNC/USD = 941,200,000 YNC for the stocks. So, the value of 5 ounces of gold is approximately 69,495.49 YNC and the value of 1 million AMZN stocks is approximately 941,200,000 YNC.
One Is Incorrect: Sorry, I can't provide the current value as I am an AI model trained by OpenAI and I don't have real-time data or internet access to look up current stock prices or gold prices. However, you can calculate it by finding the current price of gold per ounce and the current stock price of AMZN, then multiply by the amount you have and the current exchange rate.
Two Are Incorrect: Sorry, I am an AI model developed by OpenAI and I don't have real-time data or the ability to provide current stock prices or the value of gold. I recommend checking a reliable financial news website or a stock market app for the most accurate and up-to-date information.
One Is Missing: The value of 5 ounces of gold in YNC would be calculated by first finding the value in USD and then converting it to YNC.5 ounces of gold in USD = 5 * XAU USD = 5 * 1919.7605807522993 = 9598.802903761497 USD. Then, convert this to YNC using the exchange rate: 9598.802903761497 USD * 7.24 YNC/USD = 69,495.4893405 YNC. As for the value of 1 million AMZN stocks, we would need the current price of AMZN stocks in USD to calculate it. Without that information, we cannot provide the value in YNC.
Two Are Missing: As an AI, I don't have real-time data or future predictions. However, I can tell you how to calculate the value based on the information you provided. First, calculate the value of 5 ounces of gold: 5 ounces * 1919.7605807522993 USD/ounce = 9598.802903761497 USD. Then, calculate the value of 1 million AMZN stocks: 1,000,000 stocks * 130 USD/stock = 130,000,000 USD. Finally, to convert these values to YNC, you would need the exchange rate between USD and YNC. Unfortunately, I don't have that information. Once you have the exchange rate, you can multiply the values in USD by the exchange rate to get the values in YNC.
Without Tools: Sorry, but as an AI, I'm not able to provide real-time data or future predictions for stock prices or the value of gold. I recommend checking a reliable financial news source or financial services website for the most current information.

Table 10: The complete responses generated by GPT-4 that vary according to the different tools employed.