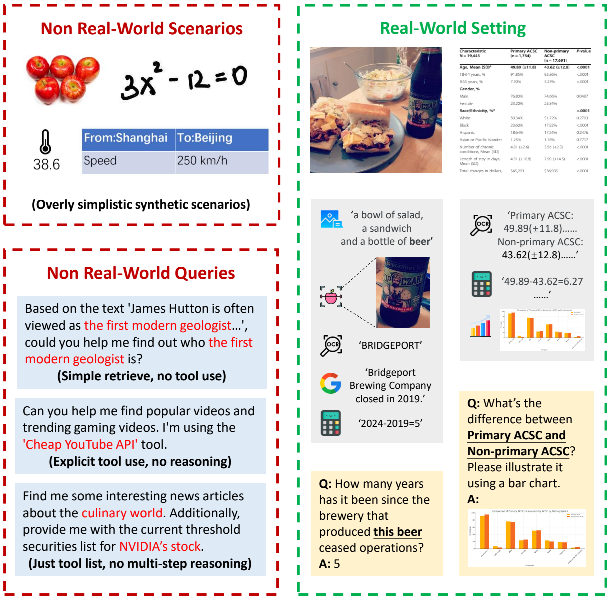
ToolVQA: A Dataset for Multi-step Reasoning VQA with External Tools

Shaofeng Yin Ting Lei Yang Liu*
Wangxuan Institute of Computer Technology, Peking University
yin_shaofeng@stu.pku.edu.cn {ting_lei, yangliu}@pku.edu.cn

Abstract

Integrating external tools into Large Foundation Models (LFMs) has emerged as a promising approach to enhance their problem-solving capabilities. While existing studies have demonstrated strong performance in tool-augmented Visual Question Answering (VQA), recent benchmarks reveal significant gaps in real-world tool-use proficiency, particularly in functionally diverse multimodal settings requiring multi-step reasoning. In this work, we introduce ToolVQA, a large-scale multimodal dataset comprising 23K samples, designed to bridge this gap. Unlike previous datasets that rely on synthetic scenarios and simplified queries, ToolVQA features real-world visual contexts and challenging implicit multi-step reasoning tasks, better aligning with real user interactions. To construct this dataset, we propose ToolEngine, a novel data generation pipeline that employs image-guided Depth-First Search (DFS) with a Longest Common Subsequence (LCS)-based example matching mechanism to simulate human-like tool-use reasoning. ToolVQA encompasses 10 multimodal tools across 7 diverse domains, with an average inference length of 2.78 reasoning steps per sample. The LLaVA-7B model fine-tuned on ToolVQA not only achieves impressive performance on the ToolVQA test set, but also surpasses the large closed-source model GPT-3.5-turbo on five out-of-distribution (OOD) datasets, showing strong generalizability in real-world tool-use scenarios. Code is available at https://github.com/Fugtemypt123/ToolVQA-release.



Based on the text 'James Hutton is often viewed as the first modern geologist…', could you help me find out who the first modern geologist is?
(Simple retrieve, no tool use)

Can you help me find popular videos and trending gaming videos. I'm using the 'Cheap YouTube API' tool.
(Explicit tool use, no reasoning)

Find me some interesting news articles about the culinary world. Additionally, provide me with the current threshold securities list for NVIDIA's stock.
(Just tool list, no multi-step reasoning)

'a bowl of salad, a sandwich and a bottle of **beer**'

'BRIDGEPORT'

'Bridgeport Brewing Company closed in 2019.'

'2024-2019=5'

Q: How many years has it been since the brewery that produced **this beer** ceased operations?
A: 5

'Primary ACSC: 49.89(±11.8)…… Non-primary ACSC: 43.62(±12.8)……'

'49.89-43.62=6.27 ……'

Q: What's the difference between **Primary ACSC and Non-primary ACSC**? Please illustrate it using a bar chart.
A:

Figure 1. Our real-world setting includes (1) complex visual scenarios with real-world context; (2) challenging queries with an implicit multi-step reasoning process. Existing datasets (left) do not meet these requirements, while our ToolVQA (right) does.

1. Introduction

Integrating external tools into Large Foundation Models (LFMs) [31] is a promising way to build versatile AI assistants. By leveraging external tools, LFMs can break down complex tasks into smaller, more manageable subtasks, each handled by a specialized tool. Prior works [7, 10, 11, 33, 37] show strong performance on traditional Visual Question Answering (VQA) benchmarks using external tools. Recent works [24, 27, 39] highlight the limitations of current LFMs in their tool-use capabilities within challenging contexts. These contexts require functionally diverse multimodal tools and multi-step reasoning processes associated with tool usage, posing major challenges to LFMs' tool-use proficiency.

One effective approach to enhancing LFM's tool-use capability is fine-tuning on large-scale datasets [8, 10, 12, 33]. However, there remains a significant gap between existing large-scale datasets and real-world user needs. Through the study of real user logs, we find that real-world tool usage involves (1) real-world scenarios: complex visual scenarios with real-world context (*e.g.* real-taken photos rather than synthetic images), and (2) real-world queries: challenging queries with an implicit multi-step reasoning process. However, existing datasets often lack these two criti-

---

*Corresponding author

cal properties, as shown in Fig. 1. The scenarios are synthesized instead of real-taken [8, 39], resulting in simplified scenes that differ from real-world scenarios in complexity. The queries require only single-step reasoning or explicitly provide hints about the reasoning process [10, 30], such as "using the Cheap YouTube API tool". These setups simplify the task and may lead to a gap between synthetic and real-world tool usage, which limits our ability to analyze LFMs' performance on this task. Additionally, some datasets [27, 39] rely on costly human annotations, making them difficult to scale, which limits fine-tuning to enhance LFMs' tool-using capabilities.

To bridge these gaps, we introduce ToolEngine, a data construction pipeline designed to generate multi-step reasoning VQA data for tool usage. ToolEngine employs an image-guided Depth-First Search (DFS) approach [30] to simulate the human-like multi-step reasoning process under the guidance of real user context examples. To facilitate multi-step reasoning in data, we introduce a Longest Common Subsequence (LCS)-based example matching mechanism. During each step in the iterative search process, this mechanism matches different real-world tool-use examples based on the previous search trajectory, guiding the controller in making the next selection. This mechanism enables: (1) Extracting rich visual information from the input image; (2) Integrating this information during the reasoning process to propose challenging queries. By freeing query generation from the constraints of fixed templates [8, 24, 30], ToolEngine enhances the utilization of visual information, consequently increasing the reasoning complexity of the generated queries.

We leverage ToolEngine to construct ToolVQA, a large-scale multimodal tool-usage dataset comprising 23K samples. We assess data quality via a manual evaluation conducted on a randomly sampled 4k subset of the training set, which yields an accuracy of 90.8%. The test set is carefully filtered and fully human-annotated. As shown in Tab. 1, ToolVQA outperforms prior tool-use datasets across multiple perspectives, providing a more comprehensive representation of real-world scenarios. We evaluate state-of-the-art (SOTA) LFMs on ToolVQA and observe that: (1) Larger LFMs demonstrate superior reasoning capabilities. (2) LFMs consistently underperform relative to humans, primarily due to their inability to integrate new information introduced in multi-turn dialogues. We fine-tune LLaVA-7B [21] on ToolVQA, and the model demonstrated performance surpassing or approaching that of the large closed-source model GPT-3.5-Turbo on both our test set and five out-of-distribution (OOD) benchmarks—TextVQA [36], TallyQA [1], Infoseek [4], TEMPLAMA [6], and GTA [39], which include unseen tasks and tools. In summary, our contributions are as follows:

1. **ToolEngine:** We introduce ToolEngine, a VQA data construction engine that generates reasonable multi-step tool-use trajectories from unannotated images, and constructs challenging question-answer pairs accordingly.
2. **ToolVQA:** We propose ToolVQA, a large-scale multimodal tool-use dataset that encompasses 10 multimodal tools across 7 diverse domains, providing a benchmark for evaluating and advancing LFMs' tool-use capabilities on real-world tasks.
3. **Fine-tuning:** We fine-tune LLaVA-7B on ToolVQA to obtain a new tool-use agent, which outperforms large closed-source models like GPT-3.5-Turbo on six benchmarks (ToolVQA and the other five OOD benchmarks).

2. Related Work

2.1. Datasets and Benchmarks for Tool Agent

With the rise of real-world tool-use applications, many works have attempted to build datasets and benchmarks to evaluate the tool-use capabilities of LFMs from different perspectives. Text-based datasets [12, 19, 29, 30, 32, 44, 47, 50] collect a large number of real-world APIs to build large-scale, diverse text-based tool-use examples. On the other hand, multimodal datasets [1, 4, 13, 24, 25, 27, 35, 36, 39, 40] integrate visual input into the query process, making queries challenging and more adaptable to real-world needs. However, they either fail to capture real-world scenarios or rely on expensive human annotation. MM-Traj [8] is a recent work which also constructs large-scale training data using an automated pipeline, but has the following limitations: (1) Synthesizes over-simplified PDF files as the multimodal contexts; (2) Their answers are long-form texts that have not been verified for correctness.

2.2. LFM-based Tool Agents

Most recent LFMs have demonstrated impressive reasoning capabilities on numerous zero-shot tasks, which makes it possible to build LFM-based tool agents for real-world tasks. Mainstream approaches include in-context learning [3, 9, 22, 37, 42, 45, 46] and instruction fine-tuning [10, 28, 30, 33, 38]. However, these agents still face challenges in real-world applications. Additionally, recent advancements in web navigation agents [14, 16, 17, 23, 41, 43, 48] have primarily focused on GUI-based interactions, often designed as browser plugins, but their applicability is limited to web-based tasks. In contrast, our ToolEngine extends to a broader range of real-world scenarios.

2.3. Other VQA Datasets

Visual Question Answering (VQA) is one of the most fundamental tasks in multimodal learning. Early VQA datasets [2, 5, 13, 18, 20, 34, 49] primarily focus on simple commonsense question-answering, which typically consists of tasks that humans can easily solve. Recently, some

Dataset	Number of Samples	Number of Tools	Mutlimodal Inputs?	Real-World S/Q Setting?	Evaluable Answers?	Reasoning Complexity↑
ToolBench [30]	126.5K	3451	✗	- / ✗	✗	1.43
ToolQA [50]	1.5K	13	✗	- / ✓	✓	1.35
Infoseek [4]	1.3M	1	✓	✓/ ✓	✓	1.13
VPD [10]*	310.5K	4	✓	✓/ ✓	-	-
M&m's [24]	1.6K	33	✓	✓/ ✗	✗	1.63
GAIA [27]†	0.5K	-	✓	✓/ ✓	✓	**2.61**
GTA [39]	0.2K	14	✓	✗/ ✓	✓	1.86
MM-Traj [8]‡	20K	9	✓	✗/ ✓	✗	1.77
ToolVQA (Ours)	23.7K	10	✓	✓/ ✓	✓	<u>2.38</u>

Table 1. Comparison between ToolVQA and other datasets. **Real-World S/Q Setting**: whether each sample in the dataset involves: (1) Scenario—utilizes real rather than synthetic multimodal inputs; (2) Query—implicitly embeds tool usage and reasoning within the query. **Reasoning Complexity**: the average reasoning depth of humans for each task. VPD* constructs trajectories from existing QA pairs and is not publicly accessible. GAIA†'s each sample corresponds to a distinct set of human-annotated tools. ‡MM-Traj generates over-simplified PDF files as multimodal contexts, and their answers are long-form texts that have not been verified for correctness.

studies [1, 4, 10, 26, 36] begin exploring more challenging forms of VQA, which require various tools like Wikipedia, OCR, Object Detection, etc. However, these datasets typically only evaluate LFM's ability to use one specific tool instead of collaboration between multiple tools, limiting LFM's multi-step reasoning ability within a tool-agent framework. In contrast, ToolVQA contains high-quality and diverse multi-step tool usage traces. It encompasses 10 multimodal tools across 7 diverse domains, with an average inference length of 2.78 reasoning steps per sample.

3. Data Collection

3.1. Task Formulation

Tool-use VQA Task. Given the image scenario $\mathcal{I}$, the toolset $\mathcal{T}$, and the user query $\mathcal{Q}$, a tool agent is required to provide an answer to the query $\mathcal{A}$ and the tool-use trajectory $\mathcal{P} = [(t_i, a_i)]_{i=1}^{n}$, in which $t_i \in \mathcal{T}$ represents the tool used in step i and a_i represents the calling arguments for t_i. Consequently, each sample is a five-tuple $(\mathcal{I}, \mathcal{T}, \mathcal{Q}, \mathcal{A}, \mathcal{P})$.

Toolset and Image Source. We collect 10 tools with various functions as our toolset, including (1) Perception: ImageCaption, OCR, ObjectDetection, and RegionDescription. (2) Operation: DrawBox, GoogleSearch. (3) Logic: Calculator, Plot, ItemCount. (4) Creativity: TextToImage. These tools cover a wide range of real-world tasks and offer strong generalization capabilities (*e.g.* GoogleSearch can retrieve any external knowledge). The different sources of image $\mathcal{I}$ and the corresponding toolset $\mathcal{T}$ are shown in Fig. 2. These sources are diverse and complex, including real-world photography, e-commerce products, school exam charts, and other practical scenarios.

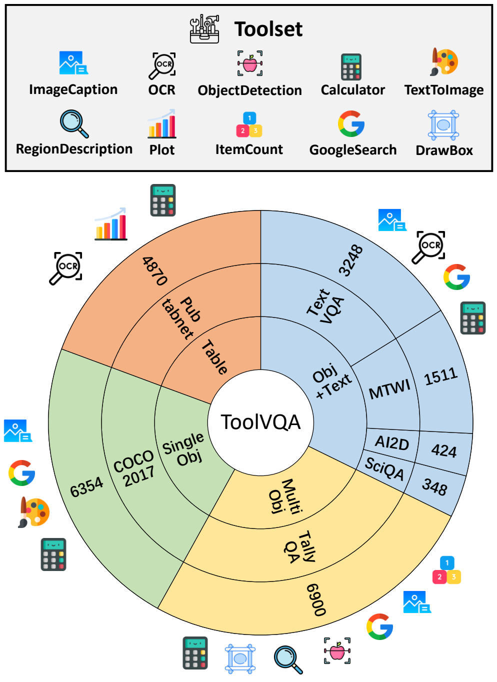


Figure 2. Image sources and corresponding tools of ToolVQA. We filter out overly simplistic tables and images from the data sources.

3.2. ToolEngine

We specify the definition of our real-world setting into the following criteria: (1) Align with human users: Each sample's scenario $\mathcal{I}$ and query $\mathcal{Q}$ should align with real-world user needs. We have filtered out uninformative scenarios $\mathcal{I}$, but it remains necessary to ensure that the tool-use queries $\mathcal{Q}$ correspond to human demands. (2) Real-world

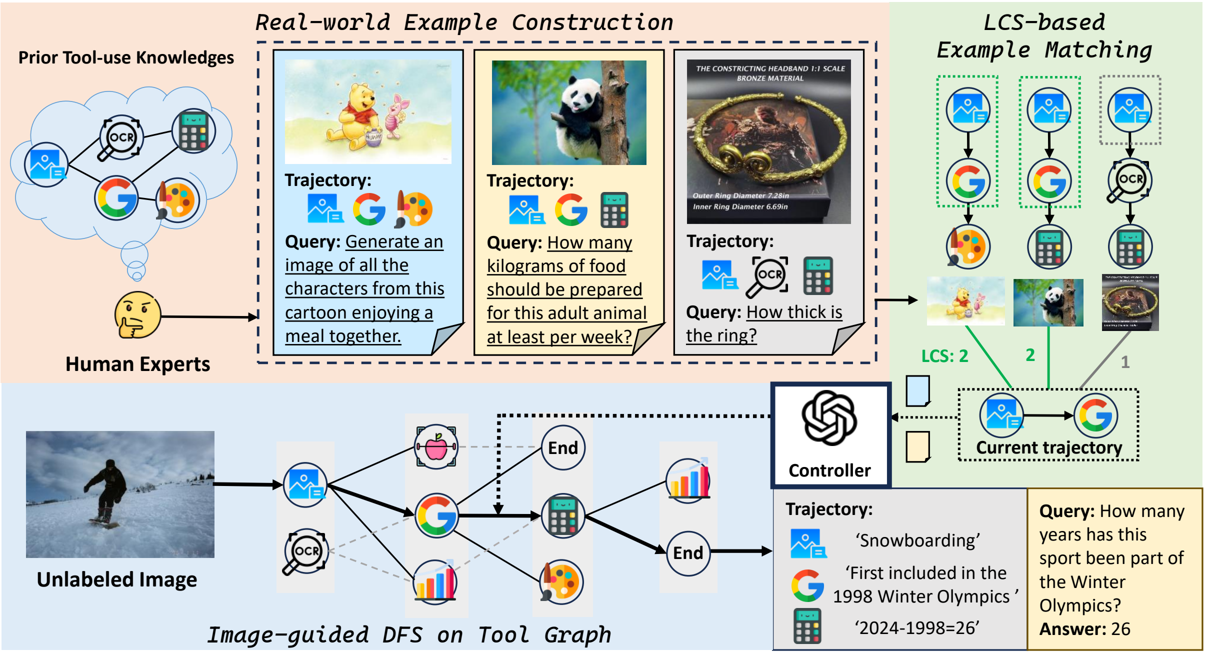


Figure 3. The pipeline of ToolEngine, contains three core components: Real-world Example Construction, Image-guided DFS on Tool Graph, and LCS-based Example Matching. Given an input image, we perform DFS on the complete tool graph. At each step, an LFM controller generates the next tool's name and arguments, guided by the image, current tool-use trajectory, and matched examples. Once DFS is complete, the tool-use trajectory is determined, and then used to generate the query and answer.

deployed tools: Real-world tool results are complex distributions with noise, and the model must learn how to deal with the noise from real results. Some work [27, 28] uses simulated tool outputs to reduce noise, but we believe this reduces the realism of the training environment. (3) Multi-step reasoning: The tool-use trajectory $\mathcal{P}$ should involve multiple steps. Each step in it must have an inherent logical connection, forming a complete problem-solving trajectory. It should not include ineffective steps (*e.g.* apply OCR to an image without text), or contain non-related tool usage in a single task (*e.g.* use "culinary world" and "NVIDIA's stock" sequentially, which should be split into two separate tasks). We propose ToolEngine, an automated data synthesis pipeline that ensures these conditions. Fig. 3 illustrates the three core components of ToolEngine:

Real-World Example Construction. To better reflect real user needs, we incorporate human prior tool-use knowledge into the LFM controller through in-context examples. Human-constructed examples provide prior knowledge about typical tool-use scenarios (*e.g.* images containing specific entities may require a search tool, while tables with multi-digit numbers may necessitate a calculator). This information is crucial for synthesizing real-world queries. Therefore, we ask human users to construct a small set of real-world examples $\mathcal{E} = \{(\mathcal{I}, \mathcal{T}, \mathcal{Q}, \mathcal{A}, \mathcal{P})\}$. Each example in this set captures a logically valid trajectory, serving as prior knowledge for automatic construction. This approach not only aligns with how humans intuitively associate tools but also provides a structured, intuitive foundation for the construction process.

Image-guided DFS on Tool Graph. To construct a multi-step tool-use trajectory, we employ image-guided Depth-First Search (DFS) on a tool graph to construct a reasonable tool-use trajectory for each image. While LFM can directly generate reasoning sequences, unrealistic tool invocations often lead to less reliable tool-use trajectories in practice. Our DFS trajectory involves real tool calls to extract detailed information from the input image, which is then used to generate challenging question-answer pairs. Specifically, we utilize the most advanced LFM (ChatGPT-4o-latest), denoted as $\mathcal{M}$, as the controller to select the tool at each step and generate the corresponding arguments. Formally:

$$t_i = \mathcal{M}(choices = \mathcal{T}, image = \mathcal{I}, examples = \text{Ret}(\mathcal{E}, \mathcal{P}_{i-1})))$$
$$a_i = \mathcal{M}(tool = t_i, image = \mathcal{I}, examples = \text{Ret}(\mathcal{E}, \mathcal{P}_i)))$$

LCS-based Example-Matching. We combine the information obtained from different tools into a reasonable reasoning trajectory to perform multi-step reasoning. Previous DFS-based methods typically match a fixed group of examples, which limited their ability to combine sufficient information for multi-step reasoning. As illustrated in Fig. 4, we

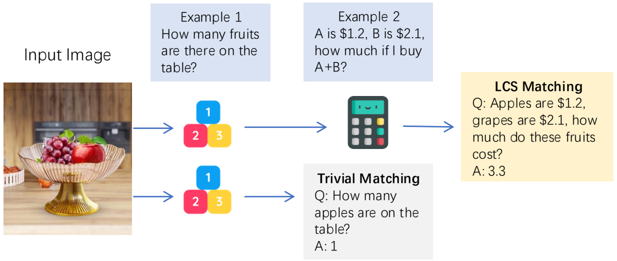


Figure 4. Comparison between different matching strategies. When a fixed example limits the diversity of generation, LCS matching can integrate multiple examples to enhance generation.

propose an example-matching method based on the Longest Common Subsequence (LCS) algorithm, which enables dynamic matching during DFS. This approach better leverages the information extraction capabilities of diverse examples, facilitating more effective multi-step reasoning. Specifically, in step i, the current solution process is denoted as $\mathcal{P}_i = [(t_1, a_1), (t_2, a_2), \cdots, (t_i, a_i)]$ and the set of example solution processes as $\mathcal{P}_e \in \mathcal{E}$. At each step, we perform the LCS matching between $\mathcal{P}_i$ and each element of $\mathcal{P}^e$, retrieving the top-k examples in $\mathcal{P}^e$ with the highest matching degree. Formally:

$$\text{Ret}(\mathcal{E}, \mathcal{P}_i) = \text{TopK}_{\mathcal{P}^e \in \mathcal{E}} \left\{ \text{LCS}(\mathcal{P}^e, \mathcal{P}_i) \right\}$$

This approach enables us to adaptively match each image with its most relevant example, while iteratively refining the match as new information is acquired at each step.

Human Annotations and Analysis. After the data synthesis process, we invite 10 human annotators to filter the generated data and re-annotate the highest-quality subset, resulting in 2,550 test samples. The initial synthetic data is estimated to have a 90.8% accuracy in human validation on 4k randomly sampled data, which is higher than that reported in previous synthetic data methods [8, 24]. Notably, the ChatGPT-4o-latest model, which we use to generate questions, demonstrates a low accuracy rate (less than 40%) in solving them. This highlights the effectiveness of our ToolEngine pipeline in decoupling complex reasoning into a series of simpler, single-step tasks. While ChatGPT-4o-latest excels in handling $\mathcal{I} \rightarrow \mathcal{P} \rightarrow \mathcal{Q}, \mathcal{A}$ process for straightforward, step-by-step reasoning, it faces challenges with the more challenging end-to-end $\mathcal{I}, \mathcal{Q} \rightarrow \mathcal{P} \rightarrow \mathcal{A}$ reasoning. This observation indicates a clear gap between LFM's single-step and multi-step reasoning capabilities.

3.3. Dataset Statistics

Tab. 2 presents the basic statistics of ToolVQA, which encompasses various question types (with both text and image as answers) and complex reasoning processes (with an average reasoning trajectory length of 2.78 steps). Notably, ToolVQA's questions and answers are concisely formulated (with an average token count of 15.74 for questions and 2.69 for answers). In contrast, previous synthetic datasets [8, 24] often fail to control answer length, leading to hallucinated content within the answers, which is detrimental to training. Fig. 5 demonstrates that our tool frequency aligns with real-world user needs, reinforcing the real-world setting of our dataset. Fig. 6 presents the distribution of reasoning trajectory lengths in ToolVQA.

3.4. Ablation on ToolEngine

To verify whether ToolEngine truly improves the quality of synthetic data, we conducted an ablation study on different synthetic methods. We invite 10 users to evaluate 1k randomly sampled data based on four key metrics: (1) answer correctness (**Acc.**); (2) correlation (**Cor.**): whether each sample's image content and Q-A pair are aligned; (3) tool redundancy (**Nec.**): whether each tool call is necessary for obtaining the answer; and (4) reasoning complexity (**R.C.**): the average reasoning depth for each task (same as defined in Tab. 1). The results, presented in Tab. 3, show that Real-World Example and LCS matching significantly improved Acc., Nec., and R.C., which highlights their role in the DFS-based simulation of reasoning chains: (1) Real-World Example and LCS matching introduce prior knowledge aligned with image content, enhancing both answer accuracy and tool necessity. (2) Real-World Example and LCS matching incorporate different types of knowledge at each step, increasing the reasoning complexity. Furthermore, the absence of LCS matching has a significant impact on data quality (*e.g*. Acc. dropped from 90.8 to 41.6), indicating that a fixed set of examples cannot adapt to different scenarios. Dynamically matching appropriate examples is crucial for significantly enhancing the generalization ability of the synthetic pipeline. However, both methods had a limited impact on Cor., likely because the existing controller already effectively captures image content, minimizing errors caused by misleading examples.

4. Experiments

4.1. Training and Test Settings

Training Objective. In our dataset $\mathcal{D}$, for each datapoint $\mathcal{E} = (\mathcal{I}, \mathcal{T}, \mathcal{Q}, \mathcal{A}, \mathcal{P})$ contains n rounds of dialogue ($n - 1$ rounds for tool usage and one round for final answer), we train our tool agent using cross-entropy loss:

$$\mathcal{L} = \mathbb{E}_{\mathcal{E} \sim \mathcal{D}} \left[\frac{1}{n} \sum_{i=1}^{n} - \log p\Big(t_i, a_i, r_i \mid \mathcal{I}, \mathcal{T}, \mathcal{Q}, \mathcal{P}_{i-1}\Big) \right]$$

Training Setting. We divide ToolVQA into training and test sets: the training set consists of 21,105 automatically generated samples (with a human verification accu-

Statistics	Number
Samples	23655
Tool Calls	65785
Text answers	15806
Image answers	7849
Query length	15.74
Answer length	2.69
Trajectory length	2.78

Table 2. Basic statistics.

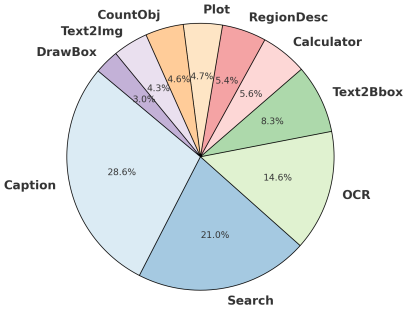


Figure 5. Tool frequency.

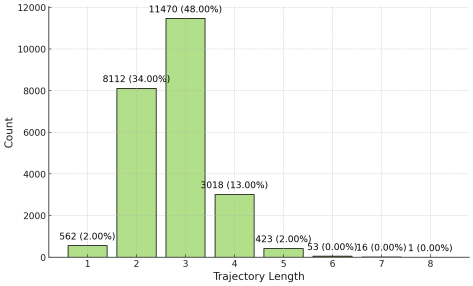


Figure 6. Trajectory length distribution.

Method	Acc.↑	Cor. ↑	Nec. ↑	R.C. ↑
ToolEngine	90.8	85.2	87.51	2.38
-w/o Example + LCS	27.3	77.6	21.04	1.1
-w/o LCS	41.6	81.4	54.26	1.61

Table 3. Ablation study on different synthetic methods.

racy of 90.8%), while the test set contains 2,550 human-reannotated samples. We fine-tune LLaVA-7B [21] in the Lego Agent framework [15].

Test Setting. We apply two test modes: end-to-end and step-by-step modes, to comprehensively evaluate LFMs' performance under different settings. The end-to-end mode only focuses on the end-to-end VQA ability of LFMs. Formally, we evaluate the $\mathcal{I} + \mathcal{Q} \rightarrow \mathcal{A}$ process. The step-by-step mode focuses on evaluating LM's tool-usage capabilities, formally the $\mathcal{I} + \mathcal{Q} + \mathcal{P}_i \rightarrow \mathcal{P}_{i+1}$ process. We use four metrics [39]: (1) InstAcc (**Inst.**): the percentage of steps executed without errors; (2) ToolAcc (**Tool.**): the accuracy of tool selection; (3) ArgAcc (**Arg.**): the accuracy of tool's argument name prediction, and (4) SummAcc (**Summ.**): the accuracy of final answer's summary. Previous work has focused on the research of large language model (LLM)-based tool agents while ignoring vision-language model (VLM)-based tool agents and the end-to-end VQA abilities of VLMs. To bridge these gaps, our test involves three settings: (1) **VLM**, directly answering questions end-to-end, formally $\mathcal{Q} + \mathcal{I} \rightarrow \mathcal{A}$, (2) **VLM+tool**, calling the tool through VLM, formally $\mathcal{Q} + \mathcal{I} + \mathcal{T} \rightarrow \mathcal{P} + \mathcal{A}$, (3) **LLM+tool**, calling the tool through LLM, formally $\mathcal{Q} + \mathcal{T} \rightarrow \mathcal{P} + \mathcal{A}$ (Note that image information can still be obtained through tools such as caption under this setting).

4.2. Main Results

Scale. As shown in Tab. 4, the tool-use capabilities of LFMs generally improve with the increase in model parameters. For example, Qwen2-7B-instruct surpasses Qwen2-2B-instruct by 7.03%, and closed-sourced LM (larger than 100B) is generally better than open-sourced (2B,7B). After fine-tuning, our 7B model's performance (last 3 lines) is close to the large scale LFM GPT-3.5-Turbo. This demonstrates that fine-tuning significantly enhances the model's real-world tool-use capabilities, even in smaller models.

Settings. By comparing the performance of the same model under different settings (VLM, VLM+tool, and LLM+tool) in Tab. 4, we identify several interesting observations. (1) In models such as GPT-4o, Qwen2-7B/2B, and LLaVA-7B, the VLM+tool setting performs worse than the LLM+tool setting, suggesting that the visual modules in current LFM architectures are still ineffective at guiding tool usage. (2) For models including GPT-4o, Claude-3.5, LLaVA-7B, and Qwen2-2B, we find that VLM+tool underperforms VLM alone. This indicates that the noise introduced by tool usage can outweigh the benefits brought by the tools. However, the tuned LLaVA model demonstrates a clear progression: VLM < LLM+tool < VLM+tool, suggesting that fine-tuning can indeed enhance the model's ability to make effective use of tools while suppressing tool-induced noise. Nevertheless, the similar performance of the LLM+tool and VLM+tool settings implies that the visual module's potential in facilitating tool use has not yet been fully explored.

Bottlenecks. The last two rows of Tab. 4 indicate that the performance of fine-tuned models is often constrained by their weakest capabilities, which are typically argument prediction and answer summarization. While fine-tuning significantly improves instruction formatting and tool selection, it makes little progress in weaker capabilities. Specifically, instruction formatting and tool selection can be accomplished through memorizing patterns, as the number of reasonable tool invocation types in real-world scenarios is limited. However, argument prediction and answer summarization require the model to genuinely understand the new information returned by tools and extract meaningful responses from it. This information varies across different scenarios and queries, and even within multi-turn dialogues of the same scenario. Consequently, a high level of generalization capability in vision-language joint reasoning is essential to handle these issues in tool-use tasks.

Model	Setting	**End-to-End Mode**			**Step-by-Step Mode**			
		Acc. (%)	Tool Call (Times)	Tool Error (Times)	Inst. (%)	Tool. (%)	Arg. (%)	Summ. (%)
Closed-source LFMs								
ChatGPT-4o-latest	VLM	**38.29**	-	-	-	-	-	-
	VLM + tool	34.96	2003	1804	36.5	14.68	8.92	56.1
	LLM + tool	37.62	4687	1962	73.02	41.12	25.27	65.45
Claude-3-5-sonnet	VLM	30.33	-	-	-	-	-	-
	VLM + tool	30.52	585	524	45.71	16.94	7.36	46.06
	LLM + tool	24.48	1834	672	66.83	39.29	13.98	75.52
GPT-3.5-Turbo	LLM + tool	18.37	3178	1321	73.24	30.46	20.08	58.18
Open-source LFMs								
Qwen2-VL-7B-Instruct	VLM	3.64	-	-	-	-	-	-
	VLM + tool	5.86	1373	1051	35.12	9.02	1.09	38.18
	LLM + tool	6.78	1366	1089	39.46	11.2	3.55	44.85
Qwen2-7B-Instruct	LLM + tool	**11.53**	1773	703	63.1	24.45	8.2	69.7
LLaVA-v1.5-7B	VLM	8.57	-	-	-	-	-	-
	VLM + tool	1.17	9684	9684	16.39	9.43	0	0.01
	LLM + tool	1.66	10421	10421	9.25	15.03	0	11.52
LLaMA-3-8b-instruct	LLM + tool	4.93	6607	6542	51.63	14.13	0.53	41.99
Qwen2-VL-2B-Instruct	VLM	**7.71**	-	-	-	-	-	-
	VLM + tool	2.1	4067	3915	47.16	12.7	0.41	21.82
	LLM + tool	2.22	6892	6655	36.34	10.52	0.68	32.12
Qwen2-1.5B-Instruct	LLM + tool	4.5	7933	6141	50.95	21.45	4.51	44.24
Tuned LLaVA-7B **(Ours)**	VLM	7.21	-	-	-	-	-	-
	VLM + tool	**18.8**	4311	617	86.62	61.61	39.34	30.91
	LLM + tool	18.43	4462	680	87.51	62.3	40.57	26.67

Table 4. Results on ToolVQA's test set.

4.3. Out-of-Distribution Benchmarks

To test the generalization capability of our tuned model, we conduct experiments on out-of-distribution (OOD) benchmarks including (1) Seen tools, unseen queries: TextVQA [36], TallyQA [1], InfoSeek [4]; and (2) Unseen tools: GTA [39], TEMPLAMA [6]. The results are shown in the Tab. 5. Our fine-tuned model significantly outperforms the baseline model LLaVA-7B by 5.8%, 4.2%, 8.6%, 21.17%, and 18.37% accuracy on these datasets, showing excellent generalizability. Its performance also surpasses the large closed-source model GPT-3.5-Turbo on unseen queries. However, its performance on TEMPLAMA is inferior to GPT-3.5-Turbo, possibly because GPT encountered similar data during its training (due to its great coverage of extensive training data), whereas our model had no exposure to such data at all.

4.4. Few-shot In-context Learning

Few-shot in-context learning (ICL) has become a common and important technique in tool-use tasks. However, prior model fine-tuning work often overlooks evaluating the ICL capabilities of fine-tuned models. We believe that fine-tuning and ICL can be complementary, offering significant potential for improvement. Tab. 6 shows few-shot ICL performance on ToolVQA. GPT-4o steadily improves with more shots, while GPT-3.5 and LLaVA both show limited gains and even decline slightly at 10-shot. This may be due to long contexts impairing the LLM's ability to interpret precise tool-use instructions. Fine-tuned LLaVA outperforms the baseline and continues to benefit from ICL. These results suggest that (1) compared to fine-tuning, ICL's improvement is limited and affected by context lengths, and (2) fine-tuned models can still benefit from ICL, highlighting the complementary value of these two approaches in enhancing tool-use performance.

4.5. Ablation Study

To further evaluate the quality of the fine-tuning data from multiple perspectives, we conduct an ablation study of the different training strategies. As shown in Tab. 7, removing the ImageCaption tool significantly degrades both accuracy and parameter correctness. This highlights the model's reliance on high-quality captions for initial scene understanding, likely due to its limited inherent visual capability for fine-grained information extraction. When all tools

Model	TextVQA [36]	TallyQA [1]	InfoSeek [4]	GTA [39]	TEMPLAMA [6]
GPT-3.5-Turbo	36.3	61	11.3	23.62	**33.67**
LLaVA-7B	41.2	60.1	5.2	12.12	3.06
Tuned LLaVA-7B	**47**	**64.3**	**13.8**	**33.29**	21.43

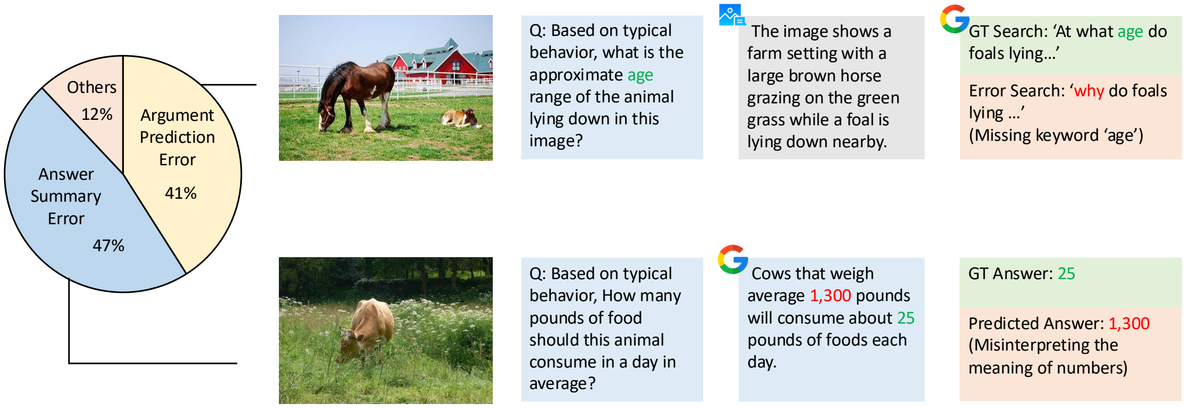
Table 5. Results on OOD benchmarks.

Model	Zero-shot	1-shot	5-shot	10-shot
GPT-4o	34.96	37.20	38.41	38.63
GPT-3.5-Turbo	18.37	19.83	20.40	19.04
LLaVA-7B	1.17	3.45	4.27	3.67
Tuned LLaVA-7B	18.80	19.41	21.13	20.69

Table 6. Few-shot performance on ToolVQA.

Model	Setting	Acc.	Tool Call	Tool Error
Tuned LLaVA-7B	VLM + tool	18.8	4311	617
-w/o Caption	VLM + tool	11.1	1861	192
-w/o $\mathcal{P}$	VLM	8.01	-	-
-w/o $\mathcal{P} + \mathcal{I}$	LLM	0.18	-	-

Table 7. Ablation study on training methods.



Q: Based on typical behavior, what is the approximate age range of the animal lying down in this image?

The image shows a farm setting with a large brown horse grazing on the green grass while a foal is lying down nearby.

GT Search: 'At what age do foals lying...'

Error Search: 'why do foals lying ...'
(Missing keyword 'age')

Q: Based on typical behavior, How many pounds of food should this animal consume in a day in average?

Cows that weigh average 1,300 pounds will consume about 25 pounds of foods each day.

GT Answer: 25

Predicted Answer: 1,300
(Misinterpreting the meaning of numbers)

Figure 7. Visualization of the main error types made by tuned LLaVA.

are removed and ToolVQA is reduced to a single-hop VQA model, performance drops substantially, underscoring the necessity of multi-step tool usage. In the absence of both tools and images, performance approaches zero, indicating that the model is heavily dependent on visual input when answering questions in ToolVQA.

4.6. Error Analysis

In Fig. 7, we randomly select 100 errors made by tuned LLaVA, and manually count and classify the main error types. The two main error types are: (1) Argument prediction error, for example, missing essential keywords "age" during a search. (2) Answer summary error, for example, extracting incorrect information "1,300" from tool outputs. The common cause of these errors is the model's failure to recognize key terms (*e.g.* "age" and "pound") in the tool outputs, even though this should not be a challenging task for a language model. This suggests that the model still struggles with dynamically adapting to additional information provided by tools. Since multi-step reasoning often suffers from error accumulation (early errors propagate to subsequent steps), errors in processing tool results at any step will affect the results. Fine-tuning has limited improvement on the capability for dynamically processing tool results, but ToolVQA still provides a valuable benchmark for evaluating this capability.

5. Conclusion

We present ToolVQA, a large multimodal dataset for real-world tool-augmented reasoning. Using our ToolEngine pipeline, we generate 23K real-world samples with implicit multi-step reasoning over 10 tools and 7 domains. This dataset challenges existing models beyond synthetic settings and better reflects real user needs. Experiments show that LLaVA-7B fine-tuned on ToolVQA not only excels on the in-domain test set but also outperforms GPT-3.5-turbo on five OOD benchmarks, establishing a new state-of-the-art for open-source models in complex tool-use VQA. ToolVQA serves as both a benchmark and a training ground for developing more capable, generalizable tool-using agents.

Acknowledgements. This work was supported by the grants from the National Natural Science Foundation of China 62372014, Beijing Nova Program and Beijing Natural Science Foundation 4252040.

References

[1] Manoj Acharya, Kushal Kafle, and Christopher Kanan. Tallyqa: Answering complex counting questions, 2018. 2, 3, 7, 8

[2] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. Vqa: Visual question answering, 2016. 2

[3] Lluis Castrejon, Thomas Mensink, Howard Zhou, Vittorio Ferrari, Andre Araujo, and Jasper Uijlings. Hammr: Hierarchical multimodal react agents for generic vqa, 2024. 2

[4] Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions?, 2023. 2, 3, 7, 8

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2

[6] Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273, 2022. 2, 7, 8

[7] Difei Gao, Lei Ji, Luowei Zhou, Kevin Qinghong Lin, Joya Chen, Zihan Fan, and Mike Zheng Shou. Assistgpt: A general multi-modal assistant that can plan, execute, inspect, and learn, 2023. 1

[8] Zhi Gao, Bofei Zhang, Pengxiang Li, Xiaojian Ma, Tao Yuan, Yue Fan, Yuwei Wu, Yunde Jia, Song-Chun Zhu, and Qing Li. Multi-modal agent tuning: Building a vlm-driven agent for efficient tool usage, 2025. 1, 2, 3, 5

[9] Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. Toolkengpt: Augmenting frozen language models with massive tools via tool embeddings, 2024. 2

[10] Yushi Hu, Otilia Stretcu, Chun-Ta Lu, Krishnamurthy Viswanathan, Kenji Hata, Enming Luo, Ranjay Krishna, and Ariel Fuxman. Visual program distillation: Distilling tools and programmatic reasoning into vision-language models, 2024. 1, 2, 3

[11] Ziniu Hu, Ahmet Iscen, Chen Sun, Kai-Wei Chang, Yizhou Sun, David A Ross, Cordelia Schmid, and Alireza Fathi. Avis: Autonomous visual information seeking with large language model agent, 2023. 1

[12] Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. Metatool benchmark for large language models: Deciding whether to use tools and which to use, 2024. 1, 2

[13] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019. 2

[14] Faria Huq, Zora Zhiruo Wang, Frank F. Xu, Tianyue Ou, Shuyan Zhou, Jeffrey P. Bigham, and Graham Neubig. Cowpilot: A framework for autonomous and human-agent collaborative web navigation, 2025. 2

[15] InternLM. Agentlego: Enhance llm agents with rich tool apis. `https://github.com/InternLM/agentlego`, 2024. 6

[16] Iat Long Iong, Xiao Liu, Yuxuan Chen, Hanyu Lai, Shuntian Yao, Pengbo Shen, Hao Yu, Yuxiao Dong, and Jie Tang. OpenWebAgent: An open toolkit to enable web agents on large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 72–81, Bangkok, Thailand, 2024. Association for Computational Linguistics. 2

[17] Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks, 2024. 2

[18] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123:32–73, 2017. 2

[19] Minghao Li, Feifan Song, Yu Bowen, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. Api-bank: A comprehensive benchmark for tool-augmented llms. In *Conference on Empirical Methods in Natural Language Processing*, 2023. 2

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 2

[21] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. 2, 6

[22] Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. In *The 37th Conference on Neural Information Processing Systems (NeurIPS)*, 2023. 2

[23] Kaixin Ma, Hongming Zhang, Hongwei Wang, Xiaoman Pan, Wenhao Yu, and Dong Yu. Laser: Llm agent with state-space exploration for web navigation, 2024. 2

[24] Zixian Ma, Weikai Huang, Jieyu Zhang, Tanmay Gupta, and Ranjay Krishna. m&m's: A benchmark to evaluate tool-use for multi-step multi-modal tasks. *EECV 2024*, 2024. 1, 2, 3, 5

[25] Arjun Mani, Nobline Yoo, Will Hinthorn, and Olga Russakovsky. Point and ask: Incorporating pointing into visual question answering, 2022. 2

[26] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge, 2019. 3

[27] Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants, 2023. 1, 2, 3, 4

[28] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023. 2, 4

[29] Yun Peng, Shuqing Li, Wenwei Gu, Yichen Li, Wenxuan Wang, Cuiyun Gao, and Michael R. Lyu. Revisiting, benchmarking and exploring api recommendation: How far are we? *IEEE Transactions on Software Engineering*, 49:1876–1897, 2021. 2

[30] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023. 2, 3

[31] Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Zhiyuan Liu, and Maosong Sun. Tool learning with foundation models, 2024. 1

[32] Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. Identifying the risks of lm agents with an lm-emulated sandbox, 2024. 2

[33] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023. 1, 2

[34] Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-okvqa: A benchmark for visual question answering using world knowledge, 2022. 2

[35] Yongliang Shen, Kaitao Song, Xu Tan, Wenqi Zhang, Kan Ren, Siyu Yuan, Weiming Lu, Dongsheng Li, and Yueting Zhuang. Taskbench: Benchmarking large language models for task automation, 2024. 2

[36] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read, 2019. 2, 3, 7, 8

[37] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11888–11898, 2023. 1, 2

[38] Chenyu Wang, Weixin Luo, Qianyu Chen, Haonan Mai, Jindi Guo, Sixun Dong, Xiaohua (Michael) Xuan, Zhengxin Li, Lin Ma, and Shenghua Gao. Mllm-tool: A multimodal large language model for tool agent learning. *arXiv preprint arXiv:2401.10727*, 2024. 2

[39] Jize Wang, Zerun Ma, Yining Li, Songyang Zhang, Cailian Chen, Kai Chen, and Xinyi Le. Gta: A benchmark for general tool agents, 2024. 1, 2, 3, 6, 7, 8

[40] Yan Wang, Yawen Zeng, Jingsheng Zheng, Xiaofen Xing, Jin Xu, and Xiangmin Xu. Videocot: A video chain-of-thought dataset with active annotation tool, 2024. 2

[41] Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory, 2024. 2

[42] Shirley Wu, Shiyu Zhao, Qian Huang, Kexin Huang, Michihiro Yasunaga, Kaidi Cao, Vassilis N. Ioannidis, Karthik Subbian, Jure Leskovec, and James Zou. Avatar: Optimizing llm agents for tool usage via contrastive reasoning, 2024. 2

[43] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024. 2

[44] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering, 2018. 2

[45] Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action, 2023. 2

[46] Zongxin Yang, Guikun Chen, Xiaodi Li, Wenguan Wang, and Yi Yang. Doraemongpt: Toward understanding dynamic scenes with large language models (exemplified as a video agent), 2024. 2

[47] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents, 2023. 2

[48] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents, 2024. 2

[49] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images, 2016. 2

[50] Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. Toolqa: A dataset for llm question answering with external tools, 2023. 2, 3