GTA: A Benchmark for General Tool Agents

Jize Wang[1,2] **Zerun Ma**[2] **Yining Li**[2] **Songyang Zhang**[2]
Cailian Chen[1] **Kai Chen**[2*] **Xinyi Le**[1*]

[1]Shanghai Jiao Tong University [2]Shanghai AI Laboratory
{jizewang2000,cailianchen,lexinyi}@sjtu.edu.cn
{mazerun,liyining,zhangsongyang,chenkai}@pjlab.org.cn

Abstract

Significant focus has been placed on integrating large language models (LLMs) with various tools in developing general-purpose agents. This poses a challenge to LLMs' tool-use capabilities. However, there are evident gaps between existing tool-use evaluations and real-world scenarios. Current evaluations often use AI-generated queries, single-step tasks, dummy tools, and text-only interactions, failing to reveal the agents' real-world problem-solving abilities effectively. To address this, we propose GTA, a benchmark for **G**eneral **T**ool **A**gents, featuring three main aspects: (i) *Real user queries*: human-written queries with simple real-world objectives but implicit tool-use, requiring the LLM to reason the suitable tools and plan the solution steps. (ii) *Real deployed tools*: an evaluation platform equipped with tools across perception, operation, logic, and creativity categories to evaluate the agents' actual task execution performance. (iii) *Real multimodal inputs*: authentic image files, such as spatial scenes, web page screenshots, tables, code snippets, and printed/handwritten materials, used as the query contexts to align with real-world scenarios closely. We design 229 real-world tasks and executable tool chains to evaluate mainstream LLMs. Our findings show that real-world user queries are challenging for existing LLMs, with GPT-4 completing less than 50% of the tasks and most LLMs achieving below 25%. This evaluation reveals the bottlenecks in the tool-use capabilities of current LLMs in real-world scenarios, which provides future direction for advancing general-purpose tool agents. Dataset and code are available at https://github.com/open-compass/GTA.

1 Introduction

Integrating tools with large language models (LLMs) has attracted broad research interest as a potential approach towards general AI assistants. Notable works include LangChain [5], AutoGPT [7], and ChatGPT Plugins [18]. These systems decompose workflow into two interactive parts: planning and execution, respectively handled by LLM controllers and callable tools. Solving complex real-world tasks requires multiple types of tools, including perception, operation, logic, and creativity, posing great challenges to LLMs' tool-use proficiency. Consequently, evaluating the models' tool-use capabilities for real-world tasks is crucial for enhancing the effectiveness of agent systems.

Despite the progress on benchmarking the tool-use capability of LLMs made by recent works, especially on collecting massive APIs and AI-generated user queries to enable scalable testing, there

*Corresponding Authors.

Workshop on Open-World Agents, 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Table 1: Comparison of benchmarks for the LLM-based agent system. *Real-world means solving the queries is helpful for humans in real life while step-implicit and tool-implicit for LLMs.

Method	Real-world* user quries	Real deployed tools	Multimodal context inputs	Human annotated tool chains	Execution result evaluation
APIBench [20]					
ToolBench [22]		✓			
APIBank [11]		✓		✓	
GAIA [15]	✓		✓		✓
m&m's [13]		✓	✓		
GTA (Ours)	✓	✓	✓	✓	✓

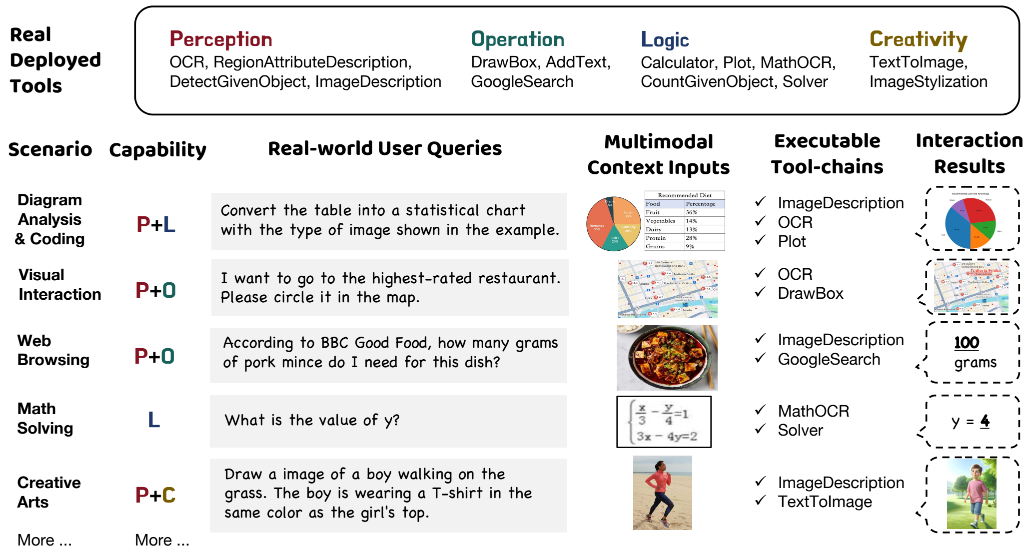


Figure 1: Some samples in the GTA benchmark. The user queries are human-designed, step-implicit, and settled in real-world scenarios. Multimodal context inputs are provided. Solving these queries is helpful for users, and complex for a LLM-based tool agent. The agent needs to use a combination of executable tools in perception, operation, logic, and creativity categories.

remain noticeable gaps regarding real-world scenarios, as shown in Table 1. First, AI-generated user queries, limited by the generative model, often result in overly brief or monotonous solutions. This is unsuitable for evaluating the reasoning and planning capability of agent systems, as shown in Table 2. Second, existing tool-use benchmarks mainly focus on text-formed user-agent interaction, lacking assessment of multimodal capabilities, thus falling short of aligning with real-world scenarios effectively. Third, existing tool-use evaluation approaches build up virtual tools. They can only evaluate isolated steps in the tool invocation chains, thus unable to reflect the agents' capability to end-to-end accomplish complex tasks.

To ensure the evaluation closely reflects real-world scenarios, we consider the authenticity of user queries, tools, and interaction modalities. We propose a comprehensive tool-use evaluation with real-world user queries. The primary features of the evaluation are:

i. ***Real user queries.*** The user queries are designed by humans, rather than generated by AI, to reflect real-world tasks accurately. These queries describe tasks with clear objectives, but the tool-use steps are implicit. Thus, the LLM must employ reasoning to deduce the suitable tools required to address the given tasks. In this way, we avoid the drawbacks of using AI-generated queries in which the tool invocation steps are often explicitly hinted at. Moreover, each query requires multiple steps to resolve, necessitating the model to plan the sequence of tool invocations.

ii. ***Real deployed tools.*** We provide an evaluation platform deployed with tools across various categories, such as perception, operation, logic, and creativity. All tools are executable rather than

Table 2: Comparison of GTA queries with AI-generated queries. The steps and tool types for queries in ToolBench and m&m's are explicitly stated, as marked in red and blue. The queries in APIBench are simple, only containing one step. Our GTA's queries are both step-implicit and tool-implicit.

Method	Queries
ToolBench	Need to create an ASCII art representation of a mathematical equation. The equation is... Help me generate the ASCII art... **Also** please **generate an ASCII art representation** of the text... (**Related tools**: figlet, list figlet styles, matheq)
APIBench	Our customer is a zoo and we want to help them **detect movement** of different animals. Write a Python program in 1 to 2 lines to call API in TensorFlowHub. (**Related tools**: ObjectDetection)
m&m's	I need an illustration for my children's book. I've imagined a scene where there's a large group of little kids... **After** we have the image, we also need to **identify all the objects**, **then** **add labels** to them. (**Related tools**: ImageGeneration, ObjectDetection, Tagging)
GTA (Ours)	Convert the table into a statistical chart with the type of image shown in the example. (**Related tools**: ImageDescription, OCR, Plot)

simulated by text description. For each task, a detailed and executable ground truth tool chain is provided, including each tool-use step and the final answer. Each step includes the tool name, argument value, and the tool return value. The detailed tool chains enable a fine-grained evaluation of the actual problem-solving abilities of tool agents.

iii. ***Real multimodal inputs.*** Each query is accompanied by one or two authentic image files, including spatial scenes, webpage screenshots, tables, code snippets, printed/handwritten materials, etc., to serve as the context for the user queries. The LLM is required to solve the problem based on the multimodal context and user queries. This setting closely aligns with the multimodal real-world problem-solving scenarios.

We manually design 229 real-world tasks and corresponding executable tool chains to evaluate mainstream LLMs. We build a platform covering a total of 14 tools across perception, operation, logic, and creation categories. Tools and some data samples are illustrated in Figure 1. We design fine-grained tool evaluation metrics that cover the entire process of tool invocation. Our findings indicate that real-world scenario queries present challenges to existing LLMs, with GPT-4 completing fewer than 50% of the tasks and the majority of LLMs managing less than 25%.

In summary, our contributions are as follows:

- A tool-use benchmark for general tool agents. The user queries are human-designed, step-implicit, and settled in real-world scenarios. Multimodal contextual inputs are provided. Each query has a corresponding executable tool chain to enable a fine-grained tool-use evaluation.
- An evaluation platform equipped with a rich variety of executable tools covering the categories of perception, operation, logic, and creativity. Fine-grained metrics are designed for tool-use, unveiling the reasoning and planning capabilities of tool-augmented LLMs in real-world scenarios.
- Evaluation and analysis of mainstream large language models. We evaluate the tool-use ability of 16 LLMs in multiple dimensions. Our findings reflect the tool-use bottleneck of existing LLMs in real-world scenarios, providing suggestions for the development path of general tool agents.

2 GTA Benchmark

In this section, we describe the design and content of GTA. The whole dataset construction pipeline is shown in Figure 2. We first present the composition of each sample in the dataset in Section 2.1. The construction method of queries and tool chains are depicted in Section 2.2 and Section 2.3, respectively. We then present the dataset's statistics in Section 2.4.

2.1 Dataset Formulation

Given a set of tools $\mathcal{T}_c = \{t_k\}_{k=1}^{N}$, a sample in GTA is composed of five parts $(\mathcal{F}, \mathcal{Q}, \mathcal{T}, \mathcal{C}, \mathcal{A})$. Among these parts, $\mathcal{F}$ is a set of files containing one or two images. $\mathcal{Q}$ is a query based on $\mathcal{F}$. It is

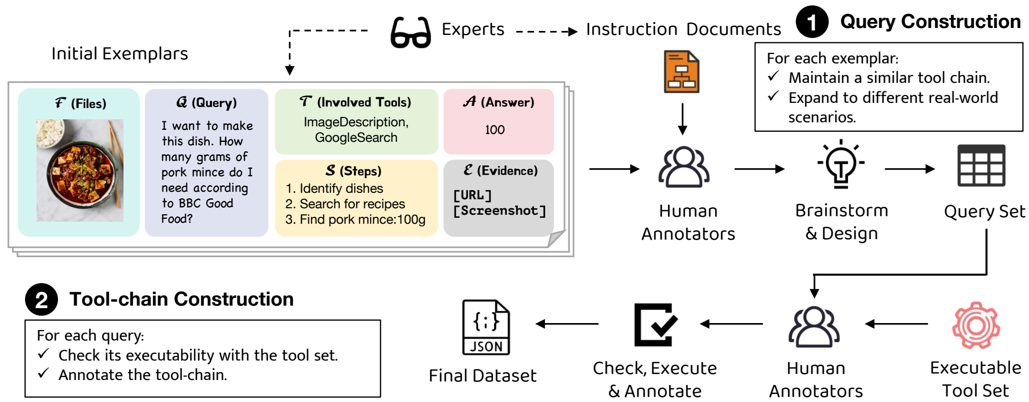


Figure 2: Two steps are performed in the dataset construction pipeline. ❶ During *query construction*, initial exemplars and instruction documents are designed by experts and given to human annotators. Annotators brainstorm and design more samples based on the exemplars. ❷ During *tool chain construction*, annotators manually call the deployed tools to check the executability of each query in the query set. Then they annotate the ground truth tool chains for each query.

a real-world scenario based problem of simple form but needs to be solved through multiple steps with tools in $\mathcal{T}_c$. Which tools need to be used, and in what steps, are not explicitly included in the query. They require reasoning and planning by the LLM, which serves as a central controller. This procedure is given in the reference tool chain $\mathcal{C} = \{s_i\}_{i=1}^m$. The tool chain contains m steps. Each step is $s_i = (t_i, a_i, r_i)$, where t_i is the tool used in step i. a_i and r_i indicate arguments and return values. $\mathcal{T} = \bigcup_{j=1}^m \{t_j\} \subseteq \mathcal{T}_c$ notes the set of tools involved in this query. $\mathcal{A}$ is the final answer yielded by the LLM after reasoning with tools.

In our setting, $\mathcal{T}_c$ contains 14 tools across four categories, including perception, operation, logic, and creativity. The full list of tools is shown in Figure 1, and more detailed information can be found in Appendix B.1. The queries $\mathcal{Q}$ are classified into three types: subjective, objective, and image generation. Examples of the three types of queries are shown in Appendix B.2. For a subjective query $\mathcal{Q}_s$, the final answer $\mathcal{A}$ is usually some descriptive text. It is not unique, but the general idea is the same. In this case, $\mathcal{A}$ contains a list of three reference answers. For an objective query $\mathcal{Q}_o$, $\mathcal{A}$ is a uniquely determined number or phrase. For an image generation query $\mathcal{Q}_g$, we do not measure the generated image directly. In this situation, $\mathcal{A} = \emptyset$.

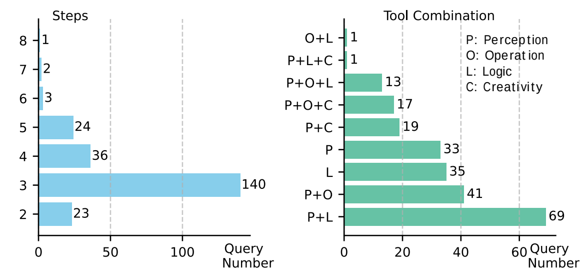
2.2 Query Construction

To construct $(\mathcal{F}, \mathcal{Q}, \mathcal{T})$, we first gather human-designed queries that meet three main principles: **i)** Given $\mathcal{T} \subseteq \mathcal{T}_c$, the task $(\mathcal{F}, \mathcal{Q})$ can be solved with the capabilities enabled by tools in $\mathcal{T}$. **ii)** To evaluate LLMs' reasoning and planning abilities, the tool invocation steps should not be explicitly stated in the queries. **iii)** The queries are meaningful and based on real-world scenarios. Satisfying all the principles simultaneously is challenging. It requires $\mathcal{F}$, $\mathcal{Q}$, and $\mathcal{T}$ to match each other in a sensible and logical way. We use a query construction pipeline based on exemplar expansion, as shown in the first part of Figure 2. We first give some initial exemplars with diverse scenarios and tool combinations. Then we instruct annotators to create more queries based on the exemplars.

Exemplar designed by experts. We first design some initial questions as exemplars, which are provided in Appendix C.1. These example questions are of diverse scenarios and contain different tool combinations. Every sample should comprise six components: $\mathcal{F}$ (image files), $\mathcal{Q}$ (queries), $\mathcal{T}$ (involved tools), $\mathcal{S}$ (solution steps), $\mathcal{A}$ (answers), and $\mathcal{E}$ (evidence). Image files $\mathcal{F}$ could be obtained from the internet and their URLs must be recorded. $\mathcal{F}$ could also be a photo taken or a diagram drawn by the annotators. The query $\mathcal{Q}$ needs to avoid obvious references to a specific tool. For example, the query *please describe the image for me* is unqualified since it obviously refers to the tool ImageDescription. The components $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{E}$, will not appear in the final dataset but are utilized to assist annotators in meeting the annotation requirements. $\mathcal{S}$ represents the steps required

Item	Number
Total query	229
Query w/ pure text answers	172
Query w/ image answers	57
Total tool calls	557
Image files	252
Tools	14
1/2/3/4-tool examples	17/147/50/15

Table 3: Basic statistics of GTA.



(a) (b)

Figure 3: Other statistics of GTA. (a) Step number per query. (b) Frequency of different tool combination.

to solve the problem. Annotators should note down the steps, ensuring their number exceeds two. The answer $\mathcal{A}$ of objective queries should be given to guarantee there is a unique answer. To ensure the uniqueness, the answer should not be dependent on the images generated in previous steps. For example, the question *what kind of animal is in the picture* should not be asked after *generate an image of an animal*, as the answer is uncertain. For queries utilizing the GoogleSearch tool, $\mathcal{E}$ should include the answer's URL and a screenshot pinpointing the answer's location to verify the query's searchability with the tool.

Diversified expansion by annotators. After the initial exemplars are given, we instruct annotators to create more samples based on each exemplar. We adopt a diversified expansion strategy for the annotators to expand the questions based on the exemplars. The general idea is to keep the tool set $\mathcal{T}$ of the template unchanged or slightly modify it. Then annotators brainstorm scenarios different from the template. Further information on the diversified expansion approach is detailed in Appendix C.2. For each sample, we have crafted a manual expansion example to serve as guidance for the annotators. After the expansion process, we perform a quality check and manually filter out the questions that do not satisfy the expansion requirements. The instruction documents for annotators are reported in Appendix C.3.

2.3 Tool Chain Construction

Based on the $(\mathcal{F}, \mathcal{Q}, \mathcal{T})$ samples constructed in Section 2.2, we instruct three annotators majoring in computer science to manually construct the corresponding tool chain $\mathcal{C}$ and the final answer $\mathcal{A}$. We design a JSON file structure, containing the query-related tool list, image paths, and ReAct [32] style dialog sequences. The dialog sequences include the user query, the executable tool chain, and the final answer. Initially, $(\mathcal{T}, \mathcal{F}, \mathcal{Q})$ are put into the associated sections for tools, images, and user queries. Subsequently, we deploy all tools in $\mathcal{T}_c$. The annotators utilize the tools according to the reference steps $\mathcal{S}$ and get the outcomes. They record this process in the tool chain section of the dialog sequences, alongside the final answer. Since we do not evaluate the tools' efficacy, when a tool fails to provide accurate recognition for a query (for instance, OCR inaccuracies in text recognition within diagrams), we discard the query. Through the above process, we ensure the feasibility of the questions, the executability of the tool chains, as well as the precision of the final answers. The structure of the tool chain is provided in Appendix C.4.

2.4 Dataset Statistics

GTA comprises a total of 229 questions, with the basic dataset statistics presented in Table 3. The dataset involves 252 images and 14 distinct tools. It includes 156 objective, 16 subjective, and 57 image-generation queries. The number of tools involved in each question varies from 1 to 4, with most questions using 2 or 3 tools. The steps to resolve the questions range from 2 to 8, with most questions requiring 2 to 4 steps, as depicted in Figure 3(a). The detailed frequency distribution of different tool combinations is listed in Figure 3(b). P, O, L, C are short for Perception, Operation, Logic, Creativity, respectively. Perception+Logic and Perception+Operation are the most frequently appearing tool combination types.

3 Evaluation and Analysis

3.1 Experiment Settings

We evaluate 16 LLMs on GTA. For API-based models, we select GPT-3.5 [19], GPT-4 [1], GPT-4o, Claude-3 [2], and Mistral-large [8]. For open-source models, we select Llama3 [14] series, Qwen1.5 [3] series, Mistral [8], Mixtral [9], Yi [33] series, Deepseek [4] series. Experiments are conducted using NVIDIA A100 GPU within OpenCompass [6] evaluation platform. We adopt Lagent [26] as the agent framework. ReAct [32] is used as the tool invocation prompt schema. More experiment information can be found in Appendix D.1 and D.2.

We evaluate the models in two modes. **Step-by-step mode** is designed to evaluate the model's fine-grained tool-use capabilities. In this mode, the model is provided with the initial n steps of the reference tool chain as prompts, with the expectation to predict the action in step $n + 1$. This method does not involve the actual use of the tool, and the prediction of each step does not depend on the model's preceding outputs. This enables an alignment comparison between the model's output with each step of the ground truth tool chain. **End-to-end mode** is designed to reflect the tool agent's actual task executing performance. In this mode, the model actually calls the tools and solves the problem by itself. Each step relies on the preceding step's output. We compare the tools selected and the execution result with the ground-truth tool set and the ground-truth result under this mode.

3.2 Evaluation Metrics

We design fine-grained metrics spanning from the LLM's tool invocation process to execution results. To evaluate the tool invocation process, we devise four metrics under step-by-step mode: ***InstAcc***, ***ToolAcc***, ***ArgAcc***, and ***SummAcc***. InstAcc is instruction-following accuracy, which quantifies the percentage of steps executed without errors. ToolAcc measures the accuracy of tool selection. ArgAcc accesses the accuracy of argument name prediction. SummAcc reflects how accurately the model can summarize the final answers considering all previous tool-use steps. For end-to-end mode, we use ***AnsAcc*** to measure the accuracy of the execution result. Besides, we calculate the ***F1 scores of tool selection*** in perception, operation, logic, and creativity categories. The four F1 scores compare the model's tool selection with the ground truth tool set, measuring its tool selection ability.

In calculating the metric AnsAcc, we exclude image generation queries and focus solely on queries with pure text answers, including subjective and objective queries. For objective queries, the ground truth contains both a whitelist and a blacklist of phrases. An answer is considered correct if it includes all terms from the whitelist and excludes all terms from the blacklist. In the case of subjective queries, the ground truth contains three manually labeled responses from distinct annotators. We compute the cosine similarity (ranging from 0 to 1) between the model's prediction and each of the three ground truth answers, ultimately considering the highest score obtained. We also design a metric ***AnsAcc w/ ImgGen***, to take image generation queries into account indirectly. Given that the outcome of the image generation is determined solely by the input parameters, we evaluate the accuracy of these parameter predictions. If the predicted parameters are correct, the images produced should align with the specified task objectives. The specific score calculation formulas of subjective and image generation queries are shown in Appendix D.4.

3.3 Main Results

Real-world tool-use tasks are challenging for existing LLMs. Current LLMs are struggling to accurately invoke tools to solve these real-world tasks. As shown in Table 4, the best-performing models, GPT-4 and GPT-4o can only correctly solve fewer than 50% of the problems, while the rest of the models solve less than 25%. This shows that real-world problems with implicit steps, real tool invocations, and multimodal contextual inputs impose high requirements on the tool-use capabilities of LLMs. Regarding model performance comparisons, API-based models outperform open-source ones. Among open-source models, Qwen1.5-72B-Chat has the highest result accuracy. Larger models within the same series perform better than their smaller counterparts, but larger models

Table 4: **Main results of GTA.** Inst., Tool., Arg., Summ., Ans., Ans.+I denote InstAcc, ToolAcc, ArgAcc SummAcc, AnsAcc, and AnsAcc w/ ImgGen respectively. P., O., L., C. denote the F1 score of tool selection in Perception, Operation, Logic, and Creativity categories. **Bold** denotes the best score among all models. Underline denotes the best score under the same model scale. **AnsAcc** reflects the overall performance.

Model	STEP-BY-STEP MODE				END-TO-END MODE					
	Inst.	Tool.	Arg.	Summ.	P.	O.	L.	C.	**Ans.**	**Ans.+I**
API-based										
GPT-4-1106-Preview	85.19	61.4	**37.88**	75	67.61	64.61	74.73	89.55	**46.59**	**44.9**
GPT-4o	**86.42**	**70.38**	35.19	72.77	**75.56**	**80**	**78.75**	82.35	41.52	40.05
GPT-3.5-Turbo	67.63	42.91	20.83	60.24	58.99	62.5	59.85	**97.3**	23.62	21.18
Claude-3-Opus	64.75	54.4	17.59	**73.81**	41.69	63.23	46.41	42.1	23.44	14.47
Mistral-Large	58.98	38.42	11.13	68.03	19.17	30.05	26.85	38.89	17.06	11.94
Open-source										
Qwen1.5-72B-Chat	48.83	24.96	7.9	68.7	12.41	11.76	21.16	5.13	13.32	10.22
Mixtral-8x7B-Instruct	28.67	12.03	0.36	54.21	2.19	34.69	37.68	42.55	9.77	9.33
Deepseek-LLM-67B-Chat	9.05	23.34	0.18	11.51	14.72	23.19	22.22	27.42	9.51	7.93
Llama-3-70B-Instruct	47.6	36.8	4.31	69.06	32.37	22.37	36.48	31.86	8.32	6.25
Yi-34B-Chat	23.73	10.77	0	34.99	11.6	11.76	12.97	5.13	3.21	2.41
Qwen1.5-14B-Chat	42.25	18.85	6.28	60.06	19.93	23.4	39.83	25.45	12.42	9.33
Qwen1.5-7B-Chat	29.77	7.36	0.18	49.38	0	13.95	16.22	36	10.56	7.93
Mistral-7B-Instruct	26.75	10.05	0	51.06	13.75	33.66	35.58	31.11	7.37	5.54
Deepseek-LLM-7B-Chat	10.56	16.16	0.18	18.27	20.81	15.22	31.3	37.29	4	3.01
Llama-3-8B-Instruct	45.95	11.31	0	36.88	19.07	23.23	29.83	42.86	3.1	2.74
Yi-6B-Chat	21.26	14.72	0	32.54	1.47	0	1.18	0	0.58	0.44

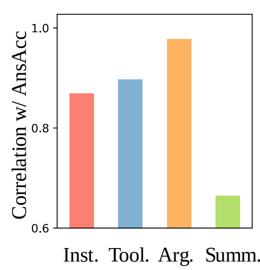


Figure 4: The Pearson correlation coefficient between AnsAcc and four metrics.

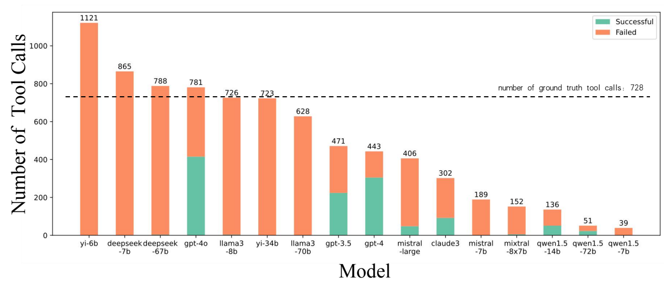


Figure 5: The number of successful and failed tool calls of each model.

from different series do not necessarily outperform the smaller ones, as shown in Figure **??**. For example, the AnsAcc of Llama3-70B-Instruct is higher than that of Llama3-8B-Instruct, but lower than Qwen1.5-7B-Chat.

The four metrics in the step-by-step mode follow the buckets effect. From the results, we observe that the overall performance of the system is affected by the lowest metric. We argue that *the four metrics in the step-by-step mode follow the buckets effect*. To verify this observation, we calculate the Pearson correlation coefficients between four metrics (InstAcc, ToolAcc, ArgAcc, SummAcc) and AnsAcc, the result is shown in Figure 4. We find that the correlation coefficient for ArgAcc with AnsAcc is the highest. ArgAcc is low for most models, indicating that the four metrics follow the buckets effect. For example, the scores of LLaMA3-70B-Instruct in InstAcc, ToolAcc, and SummAcc are higher than those of Qwen1.5-14B-Chat, but its ArgAcc is lower than Qwen1.5-14B-Chat, resulting in a lower final answer accuracy. The scores of GPT-4o in InstAcc and ToolAcc are higher than GPT-4, but its weaker argument prediction capability leads to a lower accuracy rate in the final result. The reason for the buckets effect is that under our evaluation framework, the model needs to follow user instructions, invoke tools multiple times in the correct format, and summarize the answer based on the returned results. Any error in this process can lead to an incorrect conclusion. Currently, argument prediction is the weakest capability for most models, suggesting that to enhance their general tool-use capabilities, researchers can focus on argument prediction capabilities. This concerns both the value and the format correctness of an argument.

Different series of LLMs exhibit distinct behavioral patterns. We count the number of successful and failed tool calls, illustrated in Figure 5. Successful means there are not any errors in the tool call. GPT-4o has the highest number of successful tool calls, while GPT-4 has the highest successful tool call rate. We find that models from different series exhibit distinct behavioral tendencies. Yi and Deepseek series tend to be *aggressive*, leaning towards invoking tools frequently but lacks sufficient instruction-following ability to invoke tools in a correct format. The Qwen series is *conservative*, preferring to invoke tools less often, yet it has stronger instruction-following capabilities than most other open-source models, resulting in a higher success rate of tool calls. The GPT series is *neutral*, tending to invoke tools moderately and possessing robust instruction-following abilities, which leads to the highest final answer accuracy. This suggests that to improve the performance of Yi or Deepseek, focus should be given to enhancing their instruction-following ability. Conversely, to enhance the Qwen series, reducing its conservative behavior to tool invocation could be beneficial.

Models favor either format errors or argument format errors, not both equally. We count the percentage of error types when calling tools, including format error, argument format error, and N/A (other errors, mainly containing the tools' internal error). Most models exhibit a clear tendency toward either format errors or argument format errors, rather than making both types of mistakes in nearly equal numbers. For example, Claude-3's errors are predominantly argument format-related, amounting to 82.86%, while format errors account for a mere 4.29%. This indicates that Claude-3 can follow the tool-call format well, but fails to pass the argument in a correct format.

Table 5: The percentage of different error types.

Model	Format Error (%)	Arg. Format Error (%)	N/A (%)
GPT-3.5-Turbo	8.1	60.32	20.24
GPT-4-1106-Preview	70.29	4.35	25.36
GPT-4o	78.69	19.13	13.39
Claude-3-Opus	4.29	82.86	4.29
Mistral-Large	4.47	72.07	3.07
LLaMA-3-8B-Instruct	20.47	65.15	14.38
LLaMA-3-70B-Instruct	29.51	69.7	0.8
Mistral-7B-Instruct	49.21	46.56	4.23
Mixtral-8x7B-Instruct	53.74	40.82	5.44
Qwen1.5-7B-Chat	2.56	89.74	7.69
Qwen1.5-14B-Chat	2.35	71.76	25.88
Qwen1.5-72B-Chat	10.71	71.43	17.86
Yi-6B-Chat	98.22	0.18	1.61
Yi-34B-Chat	88.11	6.22	5.67
Deepseek-LLM-7B-Chat	52.49	19.65	27.86
Deepseek-LLM-67B-Chat	58.22	34.39	7.39

4 Conclusion

We propose GTA, a real-world tool-use benchmark for general-purpose agents. The user queries are human-designed, step-implicit, and settled in real-world scenarios. Multimodal contextual inputs are provided. We build an evaluation platform equipped with executable tools in the categories of perception, operation, logic, and creation. Fine-grained metrics are designed for the tool-use capabilities of LLMs in real-world scenarios. We evaluate the tool-use capabilities of 16 LLMs. The evaluation results show that GTA is challenging for current LLMs, with advanced models like GPT-4 struggling with these real-world tasks, completing less than 50% of them. Based on our findings, we give takeaways and further suggestions on tool-use capability improvement for different models. We believe that the GTA benchmark will advance further research in identifying the model's tool-use capabilities and contribute to realizing general-purpose tool agents.

5 Limitations

Our benchmark lacks language diversity since all queries are in English. Multilingual queries can be added in future work to assess the capability of tool agents in non-English environments. Moreover, to achieve high data quality, both the user queries and the tool chains are human-written. So the cost of a data piece is higher than that of AI-generated counterparts.

6 Acknowledgements

This work is supported by the National Key R&D Program of China (No. 2022ZD0161600), and the National Natural Science Foundation of China under Grants 62422311 and 62176152.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] AI Anthropic. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 2024.

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[4] Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

[5] Harrison Chase. Langchain, October 2022.

[6] OpenCompass Contributors. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass, 2023.

[7] Significant Gravitas. Autogpt, 2023.

[8] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[9] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

[10] Chenliang Li, He Chen, Ming Yan, Weizhou Shen, Haiyang Xu, Zhikai Wu, Zhicheng Zhang, Wenmeng Zhou, Yingda Chen, Chen Cheng, et al. Modelscope-agent: Building your customizable agent system with open-source large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 566–578, 2023.

[11] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li 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, 2023.

[12] Shilong Liu, Hao Cheng, Haotian Liu, Hao Zhang, Feng Li, Tianhe Ren, Xueyan Zou, Jianwei Yang, Hang Su, Jun Zhu, et al. Llava-plus: Learning to use tools for creating multimodal agents. *arXiv preprint arXiv:2311.05437*, 2023.

[13] 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. In *Synthetic Data for Computer Vision Workshop@ CVPR 2024*, 2024.

[14] AI Meta. Introducing meta llama 3: The most capable openly available llm to date. *Meta AI Blog (accessed 2024–04–20). There is no corresponding record for this reference*, 2024.

[15] Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

[16] Yohei Nakajima. Babyagi, 2023.

[17] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[18] OpenAI. Chatgpt plugins. https://openai.com/index/chatgpt-plugins/, 2023.

[19] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[20] 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.

[21] Yujia Qin, Zihan Cai, Dian Jin, Lan Yan, Shihao Liang, Kunlun Zhu, Yankai Lin, Xu Han, Ning Ding, Huadong Wang, et al. Webcpm: Interactive web search for chinese long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8968–8988, 2023.

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

[23] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33:16857–16867, 2020.

[25] Yifan Song, Weimin Xiong, Dawei Zhu, Cheng Li, Ke Wang, Ye Tian, and Sujian Li. Restgpt: Connecting large language models with real-world applications via restful apis. *arXiv preprint arXiv:2306.06624*, 2023.

[26] Lagent Developer Team. Lagent: InternLM a lightweight open-source framework that allows users to efficiently build large language model(llm)-based agents. https://github.com/InternLM/lagent, 2023.

[27] C Wang, W Luo, Q Chen, H Mai, J Guo, S Dong, XM Xuan, Z Li, L Ma, and S Gao. Mllm-tool: A multimodal large language model for tool agent learning. *arXiv preprint arXiv:2401.10727*, 4, 2024.

[28] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023.

[29] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *arXiv preprint arXiv:2404.07972*, 2024.

[30] 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. *arXiv preprint arXiv:2303.11381*, 2023.

[31] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

[32] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

[33] Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.

[34] Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users, 2023.

[35] Wenqi Zhang, Yongliang Shen, Weiming Lu, and Yueting Zhuang. Data-copilot: Bridging billions of data and humans with autonomous workflow, 2024.

Project website: https://open-compass.github.io/GTA/.

A Related Work

LLM-based agents. In the pursuit of developing general-purpose agents, there has been considerable focus on integrating LLMs with external tools. These LLM-based agents enable powerful capabilities in environment interaction, decision-making, and task execution. Open-source platforms have been proposed, such as LangChain [5], AutoGPT [7], and BabyAGI [16]. Moreover, several efforts have been made to achieve specialized capabilities by integrating specialized tools into LLMs. WebGPT [17], WebCPM [21], WebShop [31] are proposed to enhance the model's web search ability. RestGPT [25] combines LLM with RESTful APIs to enable web service development. In the visual domain, Visual ChatGPT [28], MM-ReAct [30], MLLMtool [27], and LLaVA-Plus [12] prompt or finetune LLMs to interact with visual models. In the data analysis domain, DataCopilot [35] manages and processes massive data autonomously by invoking data analysis tools. HuggingGPT [23], ModelScopeAgent [10] build agent systems using LLMs integrated with massive machine learning models. In the field of human-computer interaction, AppAgent [34] allows LLMs to mimic human stapping and swiping operations to operate smartphones. In these works, the LLM serves as a central controller, invoking a certain class of tools to accomplish specialized tasks. In real-world scenarios, the environment is more complex. This requires LLMs to engage in planning and coordination among various types of tools, thereby posing a challenge to their tool-use capabilities.

Tool-use evaluations. With the rise of LLM-based agents, many studies have been conducted to evaluate the tool-use capabilities of LLMs. ToolBench [22] collects RESTful APIs and leverages ChatGPT [1] to design tool-use tasks and corresponding tool chains. Two metrics, Pass Rate and Win Rate, are devised to evaluate the efficacy of tool use. APIBench [20] is a comprehensive dataset that includes APIs from HuggingFace, TorchHub, and TensorHub, with evaluation metrics focusing on Abstract Syntax Tree (AST) accuracy. API-Bank [11] comprises 53 commonly utilized APIs, such as SearchEngine, PlayMusic, BookHotel, and ImageCaption, along with a comprehensive tool-augmented LLM workflow, to evaluate the API calling, retrieving, and planning abilities. m&m's [13] is a benchmark to evaluate tool-use for multi-step multimodal tasks. It aims to evaluate different planning strategies for LLMs as planning agents. Most of the aforementioned benchmarks, however, rely on AI-generated queries. The tool-use steps are explicitly and rigidly included. Thus these queries do not accurately represent real-world scenarios. Among many previous studies, GAIA [15] is renowned for its real-world scenario based benchmark aiming at evaluating general AI assistants. It evaluates more general results grounded in real-world interactions, which is closer to our work. It designs questions that are conceptually simple for humans yet challenging for most advanced AIs. However, GAIA focuses on artificial general intelligence (AGI). In contrast, GTA is designed to evaluate tool agents specifically, offering real-deployed tools and executable tool chains for a fine-grained evaluation in real-world scenarios. Osworld [29] is also a real-world benchmark featuring multi-step, complex tasks inspired by authentic user cases. Still, it is specifically tailored for computer environments, whereas GTA is devised for tool agents operating in more generalized real-world scenarios.

B Additional Information of GTA

B.1 Tool Definition

The detailed definition of 14 tools across perception, operation, logic, and creativity categories are shown in Table 6.

Table 6: Detailed definition of 14 tools across four categories.

Name	Description	Input	Output
- Perception			
OCR	Recognize the text from an image.	[image] An image containing text.	[text] The text on the image.
RegionAttributeDesc.	Describe a certain attribute of a certain part in the input image.	[image] Any image. [text] Region location and the name of attribute to describe.	[text] The description of the region.
DetectGivenObject	Detect certain object in the image.	[image] Any image. [text] Object name.	[image] An image with bounding box. [text] The location of bounding box and detecting scores.
ImageDescription	Describe the input image.	[image] Any image.	[text] The description of the image.
- Operation			
DrawBox	Draw a box on a certain location of the image.	[image] Any image. [Text] Box location.	[image] An image with a box on the certain location.
AddText	Add text on the image.	[image] Any image. [Text] Text, font size, and location.	[image] An image with text on the certain location.
GoogleSearch	Search on Google.	[text] The content to search.	[text] Searching results.
- Logic			
Calculator	Calculate by Python interpreter.	[text] Math expressions including only numbers and operation symbols.	[text] Calculation result.
Plot	Use code interpreter to draw math diagrams, statistics, etc.	[text] Python codes using Matplotlib to draw a diagram.	[image] The diagram.
MathOCR	Recognize the math expressions from a image.	[image] An image containing math expression.	[text] Latex format of the math expression.
CountGivenObject	Count the number of certain objects in the image.	[image] Any image. [text] The object name.	[text] The number of the object contained in the image.
Solver	Use code interpreter to solve math expressions.	[text] Python codes using Sympy to solve math equations or expressions containing unknown variables.	[text] Solving results.
- Creativity			
TextToImage	Generate an image from the input text.	[text] The description of an image.	[image] The image generated.
ImageStylization	Transfer the style of the image as that of a reference image.	[text] The description of the target image style. [image] An image to be transferred.	[image] The target image in the style of the text description.

B.2 Examples of Three Query Types

The examples of objective queries $\mathcal{Q}_o$, subjective queries $\mathcal{Q}_s$, and image generation queries $\mathcal{Q}_g$ are shown in Figure 6 to 11, Figure 12 to 15, and Figure 16 to 20, respectively. We provide the complete data sample, which is in the JSON format, including the involved tools, files, query, tool chain, and the final answer. To facilitate automatic evaluation, we design different final answer format for the three query types. For objective queries, the final answer contains both a whitelist and a blacklist of phrases, as shown in Figure 11. An answer is considered correct if it includes all terms from the whitelist and excludes all terms from the blacklist. In the case of subjective queries, the final answer contains three manually labeled responses from distinct annotators, as shown in Figure 15. We compute the cosine similarity (ranging from 0 to 1) between the model's prediction and each of the three ground truth answers, ultimately considering the highest score obtained. For image generation queries, the final answer is none, as shown in Figure 20, since we do not evaluate the generated images.

Query Type: Objective
Query: I need to prepare twelve servings of this dish. How many boxes of eggs will I need in total?
Involved Tools: ImageDescription, CountGivenObject, OCR
Files:

Steps:

1. Count the number of eggs in the photo.
2. Identify the eggs needed for one serving of a dish on the recipe.
3. Calculate how many eggs are needed for 12 dishes.
4. Calculate how many boxes of eggs are needed.

Answer: 2

Query and Tool Chain in JSON Format:


```
{"tools": [
    {
        "name": "ImageDescription",
        "description": "A useful tool that returns
            a brief description of the input image
           .",
        "inputs": [
            {
                "type": "image",
                "name": "image",
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            }
        ],
        "outputs": [
            {
                "type": "text",
                "name": null,
                "description": null,
```


Figure 6: (1 / 6) An example of objective query $\mathcal{Q}_o$. The final answer is a uniquely determined number or phrase.

```
                "optional": false,
                "default": null,
                "filetype": null
            }
        ]
    },
    {
        "name": "OCR",
        "description": "This tool can recognize
           all text on the input image.",
        "inputs": [
            {
                "type": "image",
                "name": "image",
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            }
        ],
        "outputs": [
            {
                "type": "text",
                "name": null,
                "description": "OCR results,
                   include bbox in x1, y1, x2, y2
                   format and the recognized
                text.",
                "optional": false,
                "default": null,
                "filetype": null
            }
        ]
    },
    {
        "name": "CountGivenObject",
        "description": "The tool can count the
           number of a certain object in the image
           .",
        "inputs": [
            {
                "type": "image",
                "name": "image",
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            },
            {
                "type": "text",
                "name": "text",
                "description": "The object
                   description in English.",
```


Figure 7: (2 / 6) An example of objective query $\mathcal{Q}_o$. The final answer is a uniquely determined number or phrase.

```
                "optional": false,
                "default": null,
                "filetype": null
            }
        ],
        "outputs": [
            {
                "type": "int",
                "name": null,
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            }
        ]
    }
],
"files": [
    {
        "type": "image",
        "path": "image/image_9.jpg",
        "url": "https://www.patersonsfruitandveg.
           co.uk/15-large_default/free-range-eggs-
           box-of-6.jpg"
    },
    {
        "type": "image",
        "path": "image/image_10.jpg",
        "url": null
    }
],
"dialogs": [
    {
        "role": "user",
        "content": " I need to prepare twelve
           servings of this dish. How many boxes of
            eggs will I need in total?"
    },
    {
        "role": "assistant",
        "tool_calls": [
            {
                "type": "function",
                "function": {
                    "name": "ImageDescription",
                    "arguments": {
                        "image": "image/image_9.
                            jpg"
                    }
                }
            }
        ],
```


Figure 8: (3 / 6) An example of objective query $\mathcal{Q}_o$. The final answer is a uniquely determined number or phrase.

```
        "thought": "Since we don't know the
           content of the two images, we use the
           image description tool for both images.
           Lets start by the first image "image/
           image_9.jpg\".\n"
    },
    {
        "role": "tool",
        "name": "ImageDescription",
        "content": {
            "type": "text",
            "content": "The image features a white
                cardboard box filled with a dozen
               eggs. The eggs are arranged in a
               neat and orderly manner, with some
               of them placed closer to the top of
               the box and others towards the
               bottom. The eggs are of various
               sizes, indicating a mix of different
                types and possibly different stages
                of development. The box is open,
               allowing the viewer to see the eggs
               clearly."
        }
    },
    {
        "role": "assistant",
        "tool_calls": [
            {
                "type": "function",
                "function": {
                    "name": "ImageDescription",
                    "arguments": {
                        "image": "image/image_10.
                           jpg"
                    }
                }
            }
        ],
        "thought": "The first image contains a box
            of eggs. Then we describe the second
           image \"image/image_10.jpg\".\n"
    },
    {
        "role": "tool",
        "name": "ImageDescription",
        "content": {
            "type": "text",
            "content": "The image features a table
                with a list of ingredients, likely
               for a recipe. The list includes
               various items such as garlic, bread,
```

Figure 9: (4 / 6) An example of objective query $\mathcal{Q}_o$. The final answer is a uniquely determined number or phrase.

```
            tomatoes, and eggs. The ingredients
               are organized in a way that makes it
                easy to follow the recipe.\nThere
               are multiple tomatoes on the table,
               with one located near the top left
               corner, another in the middle left,
               and the third one in the middle
               right. The bread is placed in the
               middle of the table"
        }
    },
    {
        "role": "assistant",
        "tool_calls": [
            {
                "type": "function",
                "function": {
                    "name": "OCR",
                    "arguments": {
                        "image": "image/image_10.
                           jpg"
                    }
                }
            }
        ],
        "thought": "The second image is likely a
           recipe. To provide an accurate answer, I
            will first need to recognize the text
           on \"image/image_10.jpg\" to understand
           the number of eggs required for a single
            serving of the dish. Then, I will count
            the number of eggs in one box. Finally,
            I can calculate the total number of
           eggs needed for twelve servings and
           determine how many boxes of eggs you
           will need in total. Let's start by
           recognizing the text on \"image/image_10
           .jpg\".\n\n"
    },
    {
        "role": "tool",
        "name": "OCR",
        "content": {
            "type": "text",
            "content": "(51, 10, 321, 73)
               Ingredients\n(81, 108, 561, 149)
               plum tomato, peeled and chopped\n
               (627, 107, 919, 148) 1 garlic clove;
                minced\n(61, 171, 565, 251) 1
               teaspoon chopped fresh basil or 1/4
               teaspoon dried basil\n(627, 172,
               1011, 209) 7 teaspoon olive oil,
               optional\n(625, 229, 1084, 269) Salt
```

Figure 10: (5 / 6) An example of objective query $\mathcal{Q}_o$. The final answer is a uniquely determined number or phrase.

```
            and pepper to taste, optional\n(63,
               275, 533, 315) 1egg or egg
               substitute equivalent\n(643, 293,
               909, 329) slice bread, toasted\n(79,
                341, 298, 373) teaspoon water\n
               (627, 351, 1044, 392) Additional
               fresh basil, optional"
        }
    },
    {
        "role": "assistant",
        "tool_calls": [
            {
                "type": "function",
                "function": {
                    "name": "CountGivenObject",
                    "arguments": {
                        "image": "image/image_9.
                           jpg",
                        "text": "egg"
                    }
                }
            }
        ],
        "thought": "The recipe shows a serving
           needs 1 egg. To calculate how many boxes
            of eggs I need, I need to know the
           number of eggs a box contains. Let's
           start by counting the number of eggs in
           \"image/image_9.jpg\""
    },
    {
        "role": "tool",
        "name": "CountGivenObject",
        "content": {
            "type": "text",
            "content": "6"
        }
    },
    {
        "role": "assistant",
        "content": "2"
    }
],
"gt_answer": {
    "whitelist": [
        [
            "2", "two"
        ]
    ],
    "blacklist": null
}
}
```


Figure 11: (6 / 6) An example of objective query $\mathcal{Q}_o$. The final answer is a uniquely determined number or phrase.

Query Type: Subjective
Query: According to the sign, what should I avoid to do now? Why?
Involved Tools: ImageDescription, OCR

Files:

Steps:

1. Recognize the image background and the icon on the sign.
2. Recognize the text in the picture.

Answer: You should avoid swimming due to the dangerous current.

Query and Tool Chain in JSON Format:


```
{"tools": [
    {
        "name": "OCR",
        "description": "This tool can recognize
           all text on the input image.",
        "inputs": [
            {
                "type": "image",
                "name": "image",
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            }
        ],
        "outputs": [
            {
                "type": "text",
                "name": null,
                "description": "OCR results,
                   include bbox in x1, y1, x2, y2
                   format and the recognized text
                   .",
                "optional": false,
                "default": null,
                "filetype": null
            }
        ]
    },
    {
        "name": "ImageDescription",
        "description": "A useful tool that returns
            a brief description of the input image
           .",
        "inputs": [
            {
                "type": "image",
                "name": "image",
                "description": null,
                "optional": false,
```


Figure 12: (1 / 4) An example of subjective query $\mathcal{Q}_s$. The final answer is usually some descriptive text. It is not unique, but the general idea is the same.

```
                "default": null,
                "filetype": null
            }
        ],
        "outputs": [
            {
                "type": "text",
                "name": null,
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            }
        ]
    }
],
"files": [
    {
        "type": "image",
        "path": "image/image_27.jpg",
        "url": "https://images.sbs.com.au/dims4/
           default/8a59a3f/2147483647/strip/true/
           crop/3000x1688+0+0/resize/1280x720!/
           quality/90/?url=http%3A%2F%2Fsbs-au-
           brightspot.s3.amazonaws.com%2Fe0%2F64%2
           F10b664a04c59a03b066c4254880e%2Fbeach-
           safety-sign-cropped.jpg&imwidth=1280"
    }
],
"dialogs": [
    {
        "role": "user",
        "content": "According to the sign, what
           should I avoid to do now? Why?"
    },
    {
        "role": "assistant",
        "tool_calls": [
            {
                "type": "function",
                "function": {
                    "name": "ImageDescription",
                    "arguments": {
                        "image": "image/image_27.
                           jpg"
                    }
                }
            }
        ],
        "thought": "I should first describe the
           image for detailed information."
    },
```

Figure 13: (2 / 4) An example of subjective query $\mathcal{Q}_s$. The final answer is usually some descriptive text. It is not unique, but the general idea is the same.

```
    {
        "role": "tool",
        "name": "ImageDescription",
        "content": {
            "type": "text",
            "content": "The image features a
               person standing on a beach, with a
               yellow sign in the foreground. The
               sign is placed on the sand, and it
               appears to be a warning or a caution
                sign. The person is walking past
               the sign, possibly enjoying the
               beach atmosphere. The scene captures
                a moment of leisure and outdoor
               activity."
        }
    },
    {
        "role": "assistant",
        "tool_calls": [
            {
                "type": "function",
                "function": {
                    "name": "OCR",
                    "arguments": {
                        "image": "image/image_27.
                           jpg"
                    }
                }
            }
        ],
        "thought": "There seems a sign in the
           image. I will extract all text on the
           sign."
    },
    {
        "role": "tool",
        "name": "OCR",
        "content": {
            "type": "text",
            "content": "(428, 118, 929, 603) X
               DANGEROUS CURRENT"
        }
    },
    {
        "role": "assistant",
        "content": "You should avoid going into
           the water now due to the dangerous
           current."
    }
],
```

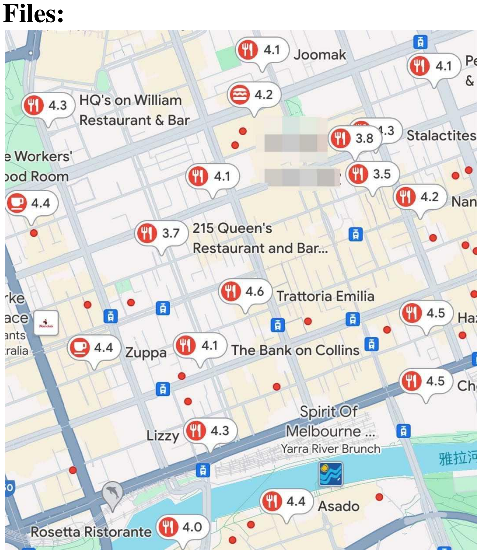

Figure 14: (3 / 4) An example of subjective query $\mathcal{Q}_s$. The final answer is usually some descriptive text. It is not unique, but the general idea is the same.

```
"gt_answer": [
    "You should avoid swimming. Because there is
       dangerous current.",
    "You should avoid swimming in the sea because
       the signs in the picture indicate that it is
        a dangerous area and swimming can be
       dangerous.",
    "According to the sign, I should avoid to go
       swimming in the sea. The background of the
       picture is a sea area, and there is a yellow
        warning sign with 'DANGEROUS CURRENT'
       written on it. Additionally, a red cross is
       marked over the act of swimming, indicating
       that swimming here is dangerous and
       prohibited. Therefore, I should avoid
       swimming in the sea."
]
}
```

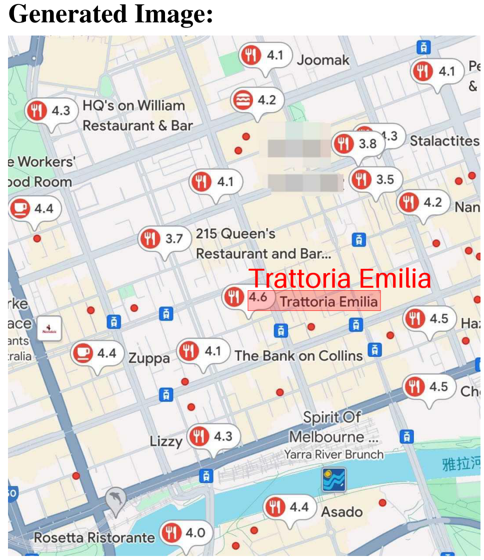
Figure 15: (4 / 4) An example of subjective query $\mathcal{Q}_s$. The final answer is usually some descriptive text. It is not unique, but the general idea is the same.

Query Type: Image Generation
Query: I want to go to the highest-rated restaurant. Please circle it in the map.
Involved Tools: OCR, DrawBox

Files:



Generated Image:



Steps:

1. Identify the ratings of each restaurant in the map using OCR tool.
2. Identify the restaurant with the highest rating and its coordinate from the OCR result.
3. Circle the restaurant in the graph using DrawBox tool.

Figure 16: (1 / 5) An example of image generation query $\mathcal{Q}_i$. The final answer is none since we do not evaluate the generated image directly.

Query and Tool Chain in JSON Format:


```
{"tools": [
    {
        "name": "OCR",
        "description": "This tool can recognize
          all text on the input image.",
        "inputs": [
            {
                "type": "image",
                "name": "image",
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            }
        ],
        "outputs": [
            {
                "type": "text",
                "name": null,
                "description": "OCR results,
                   include bbox in x1, y1, x2, y2
                   format and the recognized text
                   .",
                "optional": false,
                "default": null,
                "filetype": null
            }
        ]
    },
    {
        "name": "DrawBox",
        "description": "A tool to draw a box on a
          certain region of the input image.",
        "inputs": [
            {
                "type": "image",
                "name": "image",
                "description": null,
                "optional": false,
                "default": null,
                "filetype": null
            },
            {
                "type": "text",
                "name": "bbox",
                "description": "The bbox
                   coordinate in the format of `(x1
                   , y1, x2, y2)`",
                "optional": false,
                "default": null,
                "filetype": null
            },
```


Figure 17: (2 / 5) An example of image generation query $\mathcal{Q}_i$. The final answer is none since we do not evaluate the generated image directly.

```
                {
                    "type": "text",
                    "name": "annotation",
                    "description": "The extra
                       annotation text of the bbox",
                    "optional": true,
                    "default": null,
                    "filetype": null
                }
            ],
            "outputs": [
                {
                    "type": "image",
                    "name": null,
                    "description": null,
                    "optional": false,
                    "default": null,
                    "filetype": null
                }
            ]
        }
    ],
    "files": [
        {
            "type": "image",
            "path": "image/image_129.jpg",
            "url": null
        }
    ],
    "dialogs": [
        {
            "role": "user",
            "content": "I want to go to the highest-
               rated restaurant.Please circle it in the
                map."
        },
        {
            "role": "assistant",
            "tool_calls": [
                {
                    "type": "function",
                    "function": {
                        "name": "OCR",
                        "arguments": {
                            "image": "image/image_129.
                               jpg"
                        }
                    }
                }
            ],
```


Figure 18: (3 / 5) An example of image generation query $\mathcal{Q}_i$. The final answer is none since we do not evaluate the generated image directly.

```
        "thought": "To identify the highest-rated
           restaurant on the map provided in the
           image file, I will first need to use the
            OCR tool to recognize any text on the
           image, which might include restaurant
           names and their ratings. After
           identifying the highest-rated restaurant
           , I can then use the DrawBox tool to
           circle it on the map.\n\n"
    },
    {
        "role": "tool",
        "name": "OCR",
        "content": {
            "type": "text",
            "content": "(588, 30, 787, 75) 4.1
               Joomak\\n(982, 54, 1074, 98) P 4.1\\
               n(98, 141, 430, 222) 4.3 HQ’s on
               William Restaurant & Bar\\n(568,
               132, 616, 164) 4.2\\n(798, 215,
               1080, 264) 3 3.8 Stalactites\\n(0,
               270, 155, 353) 8 Workers’ od Room\\n
               (476, 318, 516, 348) 4.1\\n(838,
               314, 886, 342) 3.5\\n(58, 380, 108,
               412) 4.4\\n(946, 366, 1080, 409) 4.2
                Nan\\n(356, 435, 745, 519) 3.7 215
               Queen’s Restaurant and Bar_\\n(0,
               595, 60, 748) rke ace ants ralia\\n
               (548, 582, 851, 628) 4.6 Trattoria
               Emilia\\n(962, 630, 1080, 674) 4.5
               Haz\\n(615, 663, 633, 687) 0\\n(202,
                710, 377, 766) 4.4 Zuppa\\n(446,
               704, 817, 751) 4.1 The Bank on
               Collins\\n(960, 788, 1080, 831) 4.5
               Ch\\n(627, 851, 865, 975) Spirit Of
               Melbourne Yarra River Brunch\\n(468,
                900, 518, 932) 4.3\\n(982, 950,
               1076, 1004) iti;\\n(0, 1035, 21,
               1055) 0\\n(642, 1062, 817, 1107) 4.4
                Asado\\n(55, 1128, 343, 1165)
               Rosetta Ristorante\\n(402, 1120,
               454, 1150) 4.0\\n(315, 904, 407,
               952) Lizzy\\"
        }
    },
    {
        "role": "assistant",
        "tool_calls": [
            {
                "type": "function",
```

Figure 19: (4 / 5) An example of image generation query $\mathcal{Q}_i$. The final answer is none since we do not evaluate the generated image directly.

```
                "function": {
                    "name": "DrawBox",
                    "arguments": {
                        "bbox": "(548, 582, 851,
                            628)"
                    }
                }
            }
        ],
        "content": "Based on the OCR results, the
            highest-rated restaurant identified is
            \"Trattoria Emilia\" with a rating of
            4.6."
    },
    {
        "role": "tool",
        "name": "DrawBox",
        "content": {
            "type": "image",
            "content": "image/
                dummy_generated_image.jpg"
        }
    },
    {
        "role": "assistant",
        "content": "The image is saved."
    }
],
"gt_answer": null
}
```


Figure 20: (5 / 5) An example of image generation query $\mathcal{Q}_i$. The final answer is none since we do not evaluate the generated image directly.

C Additional Information for Data Design

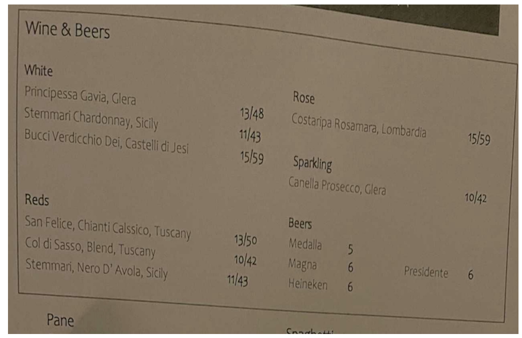
C.1 Query Exemplars

We design several initial queries as query exemplars, as shown from Figure 21 to 35. The annotators brainstorm and design new questions that have the same tool chain as the exemplar but with different scenarios. We provide an expansion example for most exemplars for annotators to refer to.

Exemplar 1

Query: How much should I pay for the beer on the table according to the price on the menu?
Involved Tools: ImageDescription, CountGivenObject, OCR, Calculator

Files:



Steps:

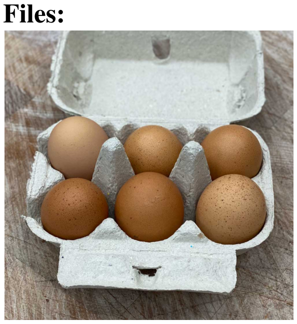
1. Count the number of beers.
2. Recognize text on the bottles.
3. Recognize text on the menu.
4. Calculate the total price of the beers.

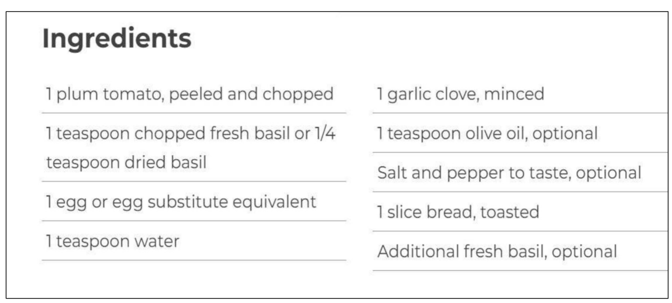
Answer: 12

Expansion Example

Query: I need to prepare twelve servings of this dish. How many boxes of eggs will I need in total?
Involved Tools: ImageDescription, CountGivenObject, OCR, Calculator

Files:





Ingredients

1 plum tomato, peeled and chopped	1 garlic clove, minced
1 teaspoon chopped fresh basil or 1/4 teaspoon dried basil	1 teaspoon olive oil, optional
1 egg or egg substitute equivalent	Salt and pepper to taste, optional
1 teaspoon water	1 slice bread, toasted
	Additional fresh basil, optional

Steps:

1. Count the number of eggs in the photo.
2. Identify the eggs needed for one serving of a dish on the recipe.
3. Calculate how many eggs are needed for 12 dishes.
4. Calculate how many boxes of eggs are needed.

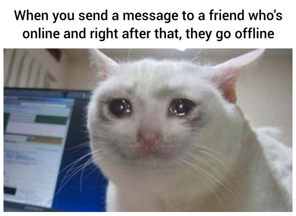
Answer: 2

Figure 21: Query exemplar 1.

Exemplar 2

Query: Can you explain this meme?
Involved Tools: OCR, ImageDescription

Files:



Steps:

1. Recognize the text in the picture.
2. Describe the content of the image.
3. Infer the central idea in relation to the image and the text.

Answer: The meme shows it is sad when we send a message to a friend who's online and right after that, they go offline. It's a coincidental and unpleasant situation.

Expansion Example

Query: What sports event was this photo taken at? Please provide the names of the two opposing teams in your answer.
Involved Tools: OCR, ImageDescription

Files:

Steps:

1. Identify the words in the picture: Lakers, Suns.
2. Describe the content of the picture: basketball game.

Answer: Lakers vs suns basketball game.

Figure 22: Query exemplar 2.

Exemplar 3

Query: What is the woman in a pink shirt doing?
Involved Tools: DetectGivenObject, RegionAttributeDescription
Files:

Steps:

1. Detect the woman in pink.
2. Describe the action of the person in the detection box.

Answer: Serving food.

Expansion Example

Query: What is the breed of the dog in the middle of the picture?
Involved Tools: DetectGivenObject, RegionAttributeDescription

Files:

Steps:

1. Detect all the dogs.
2. Find the detection box in the center.
3. Describe the dog's breed in the detection box.

Answer: Corgi.

Figure 23: Query exemplar 3.

Exemplar 4

Query: What is x in the equation?
Involved Tools: MathOCR, Solver

Files:

$$(x+3)^2=4$$

Steps:

1. Convert the handwritten image into latex style.
2. Solve the equation.

Answer: -1 or -5.

Expansion Example

Query: What is the image of this analytic formula?
Involved Tools: MathOCR, Plot

Files:

$$y = x^2 + 2x - 1$$

Steps:

1. Convert the handwritten image into latex style.
2. Plot according to the math expression.

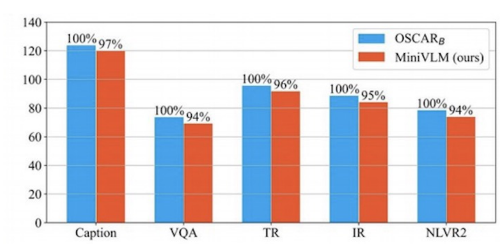
Figure 24: Query exemplar 4.

Exemplar 5

Query: Convert the table into a statistical chart with the type of image shown in the example. The horizontal axis is the country, and the vertical axis uses three colors for sales volume, revenue, and profit.
Involved Tools: ImageDescription, OCR, Plot

Files:

Country	Sales Volume	Revenue	Profit
USA	40.080	$15.971.880	$3.086.421
China	35.070	$15.866.670	$3.032.162
Australia	27.054	$14.812.566	$2.868.636
India	23.046	$10.608.174	$1.853.710
South Korea	16.032	$10.494.948	$1.975.844



Steps:

1. Recognize text in the table.
2. Describe the style of the statistical chart.
3. Plot the diagram in the same style with the data from the table.

Figure 25: Query exemplar 5.

Exemplar 6

Query: What percentage of people wear helmets?
Involved Tools: DetectGivenObject, RegionAttributeDescription, Calculator
Files:

Steps:

1. Detect all the people.
2. Describe each of the people whether he wears a helmet.
3. Calculate the percentage.

Answer: 62.5%.

Expansion Example

Query: What's the total number of the mother swans and the baby swans?
Involved Tools: CountGivenObject, ImageDescription, Calculator

Files:

Steps:

1. Count the number of mother swans.
2. Count the number of baby swans.
2. Calculate the total number.

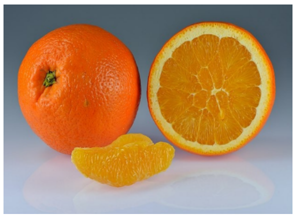
Answer: 7.

Figure 26: Query exemplar 6.

Exemplar 7

Query: I'm a 23-year-old female. How many grams of this kind fruit can I meet the vitamin C intake recommended by U.S. Recommended Dietary Allowance in 2021? Please round your answers to the nearest gram. You can look for information in National Institutes of Health and Wikipedia.
Involved Tools: ImageDescription, GoogleSearch, Calculator

Files:



Steps:

1. Identify the fruit in the picture as an orange.
2. Search Wikipedia for the VC content of oranges: 53mg/100g.
3. Search National Institutes of Health's recommended VC intake for adults: 75mg for women, 90mg for men.
4. Calculate the intake of oranges = recommended VC intake (I'm a woman, take 75mg)/VC content, and round it up.

Answer: 142.

Evidence:
https://en.wikipedia.org/wiki/Vitamin_C
https://ods.od.nih.gov/factsheets/VitaminC-HealthProfessional/

Adverse effects
Diet
Sources
Pharmacology
Chemistry
Testing
Synthesis
History
Society and culture
Pharmacopoeias
Notes
References
External links

Raw plant source[96]	Amount (mg / 100g)
Kakadu plum	1000–5300[97]
Camu camu	2800[95][98]
Acerola	1677[99]
Indian gooseberry	445[100][101]
Rose hip	426
Common sea-buckthorn	400[102]
Guava	228
Blackcurrant	200
Yellow bell pepper/capsicum	183
Red bell pepper/capsicum	128
Kale	120
Broccoli	90
Kiwifruit	90

Raw plant source[96]	Amount (mg / 100g)
Green bell pepper/capsicum	80
Brussels sprouts	80
Loganberry, redcurrant	80
Cloudberry, elderberry	60
Strawberry	60
Papaya	60
Orange, lemon	53
Cauliflower	48
Pineapple	48
Cantaloupe	40
Passion fruit, raspberry	30
Grapefruit, lime	30
Cabbage, spinach	30

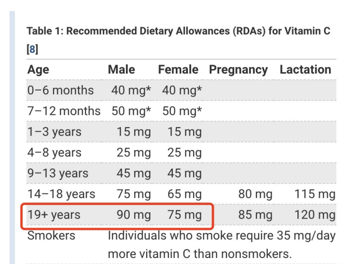


Table 1: Recommended Dietary Allowances (RDAs) for Vitamin C [8]

Age	Male	Female	Pregnancy	Lactation
0–6 months	40 mg*	40 mg*		
7–12 months	50 mg*	50 mg*		
1–3 years	15 mg	15 mg		
4–8 years	25 mg	25 mg		
9–13 years	45 mg	45 mg		
14–18 years	75 mg	65 mg	80 mg	115 mg
19+ years	90 mg	75 mg	85 mg	120 mg
Smokers	Individuals who smoke require 35 mg/day more vitamin C than nonsmokers.			

Expansion Example

Query: According to Midwest Dairy, how many gallons of milk can this animal produce at most in 725 days?
Involved Tools: ImageDescription, GoogleSearch, Calculator

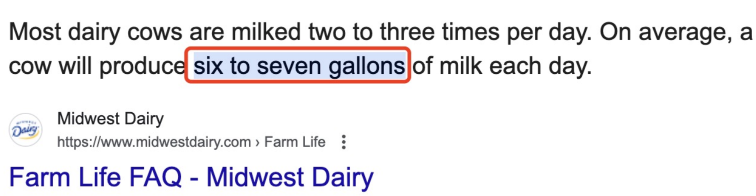
Files:

Steps:

1. Identify the animal in the image as a dairy cow.
2. Search for the average daily milk production for cows recorded on Midwest Dairy: 6-7 gallons.
3. Calculate the maximum production over a 725 day period: 725*7.

Answer: 5075.

Evidence:
https://www.midwestdairy.com/farm-life/farm-life-faq/



Most dairy cows are milked two to three times per day. On average, a cow will produce six to seven gallons of milk each day.

Midwest Dairy
https://www.midwestdairy.com › Farm Life
Farm Life FAQ - Midwest Dairy

Figure 27: Query exemplar 7.

Exemplar 8

Query: How much did I spend on food totally?
Involved Tools: OCR, Calculator
Files:

Steps:

1. Identify goods and their prices.
2. Identify the food in the bill.
3. Calculate the total price of the food.

Answer: 10.81

Expansion Example

Query: We are a family of 5 and everyone takes fish oil. How many days is this bottle of fish oil enough for us?
Involved Tools: OCR, Calculator

Files:

Steps:

1. Identify key information from the bottle: 1 per day, 290 softgels.
2. Calculate the bottle number: 290/5.

Answer: 58

Figure 28: Query exemplar 8.

Exemplar 9

Query: I have 22 dollars. For lunch, my mom and I would each like an entree and a dessert. I don't eat doughnuts and my mom doesn't eat chicken. All of our food should be different. What specific foods can I buy?

Involved Tools: OCR, Calculator

Files:

Steps:

1. Identify dishes and prices.
2. Find the food that meets the constraints.
3. Find out the food with total price less than $22.

Answer: For you, a Chicken Burger for the entree and a Pan-Cake for the dessert. For your mom, a Beef Burger for the entree and a Jelly Doughnuts for the dessert.

Expansion Example

Query: I need a total ethereum hash rate of at least 122 MH/s, and the total rated power should not exceed 510 W. Which two GPU should I buy?

Involved Tools: OCR, Calculator

Files:

SPECIFICATIONS

	30HX	40HX	50HX	90HX
Ethereum Hash Rate[1]	26 MH/s	36 MH/s	45 MH/s	86 MH/s
Rated Power[2]	125 W	185 W	250 W	320 W
Power Connectors[2]	1x 8-pin	1x 8-pin	2x 8-pin	2x 8-pin
Memory Size	6GB	8GB	10GB	10GB
Starting Availability	Q1	Q1	Q2	Q2

[1] Ethereum hash rate applies to the DAG and algorithm in use in Epoch 394 and is provided for reference clocks under room temperature conditions with good cooling.
[2] Rated Power and Power connectors specified for the reference design. Partner board designs may choose a different configuration.

Steps:

1. Identify GPUs and their prices.
2. Find out GPUs with summed power greater than 122MH/s and less than 510W.

Answer: One 40HX and one 90HX.

Figure 29: Query exemplar 9.

Exemplar 10

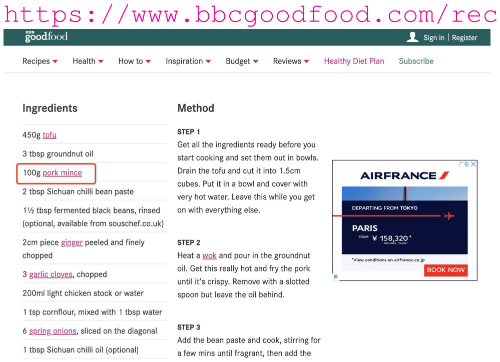
Query: I want to make this dish. How many grams of pork mince do I need according to BBC Good Food?
Involved Tools: ImageDescription, GoogleSearch
Files:

Steps:

1. Identify the dish.
2. Search BBC Good Food for recipes and ingredient lists.
3. Find out the gram number of pork mince.

Answer: 100

Evidence:
https://www.bbcgoodfood.com/recipes/mapo-tofu


goodfood Sign in | Register

Recipes How to Health Inspiration Budget Reviews Healthy Diet Plan Subscribe

Ingredients

- 450g tofu
- 3 tbsp groundnut oil
- 100g pork mince
- 2 tbsp Sichuan chilli bean paste
- 1½ tbsp fermented black beans, rinsed (optional, available from souschef.co.uk)
- 2cm piece ginger peeled and finely chopped
- 3 garlic cloves, chopped
- 200ml light chicken stock or water
- 1 tsp cornflour, mixed with 1 tbsp water
- 6 spring onions, sliced on the diagonal
- 1 tbsp Sichuan chilli oil (optional)

Method

STEP 1
Get all the ingredients ready before you start cooking and set them out in bowls. Drain the tofu and cut it into 1.5cm cubes. Put it in a bowl and cover with very hot water. Leave this while you get on with everything else.

STEP 2
Heat a wok and pour in the groundnut oil. Get this really hot and fry the pork until it's crispy. Remove with a slotted spoon but leave the oil behind.

STEP 3
Add the bean paste and cook, stirring for a few mins until fragrant, then add the

Expansion Example

Query: I want to go to this place in Shanghai, place tell me it's "Regular" ticket price in June, 2023. Please answer in RMB.
Involved Tools: ImageDescription, GoogleSearch

Files:

Steps:

1. Identify the building in the picture.
2. Search for "Regular" ticket price for Shanghai Disney in 2023.

Answer: 475

Evidence:
https://www.shanghaidisneyresort.com/en/new-pricing-structure/


Pricing Rate Adjustment for Shanghai Disneyland Admission Effective June 23, 2023

Publication Date: December 23, 2022
Shanghai International Theme Park and Resort Management Company Limited

The following is a public notice of pricing rate adjustment for Shanghai Disneyland admission, which will take effect on June 23, 2023.

Starting June 23, 2023, Shanghai Disney Resort will adjust its pricing rate for admission to Shanghai Disneyland. The new definition and rate for the four-tiered pricing structure – Regular, Regular Plus, Peak and Peak Plus, will be as following:

- **"Regular"** price of admission to Shanghai Disneyland, covering most of the weekdays and selected weekends, is set at 475 RMB;
- **"Regular Plus"** price of admission will cover selected weekends and selected weekdays, and is set at 599 RMB;
- **"Peak"** price of admission, which covers most of the days in summer season, selected Chinese statutory holiday periods (including their shoulder days), internationally recognized celebration periods, park special event days, and other peak visitation days, is set at 719 RMB;
- **"Peak Plus"** price of admission, covering selected Chinese statutory holiday periods, park special event days and selected days in summer season, is set at 799 RMB.

Figure 30: Query exemplar 10.

Exemplar 11

Query: I will get off work at 5:00 today. I need to spend an hour for dinner and half an hour to get to the movie theater. Which is the earliest movie show I can catch? Please circle it in the screenshot.
Involved Tools: OCR, DrawBox
Files:

MONDAY 1/15/24

TIME	TITLE
5:00am	The Little Princess (1939)
Featuring: Shirley Temple, Richard Greene	
7:00am	A Room With A View (1985)
Featuring: Maggie Smith, Helena Bonham Carter	
9:35am	The Trip To Bountiful (1985)
Featuring: Geraldine Page, John Heard	
11:55am	Cinderella Liberty (1973)
Featuring: James Cann, Marsha Mason, Kirk Calloway	
2:25pm	Rough Magic (1995)
Featuring: Bridget Fonda, Russell Crowe	
4:40pm	Friends with Kids (2011)
Featuring: Adam Scott, Jennifer Westfeldt	
7:00pm	A Walk To Remember (2002)
Featuring: Mandy Moore, Shane West	
9:10pm	If Only (2004)
Featuring: Jennifer Love Hewitt, Paul Nicholls	
11:15pm	Across the Tracks (1990)
Featuring: Brad Pitt, Ricky Schroder	
1:25am	Rock 'N' Roll High School (1979)
Featuring: P.J. Soles, Vincent Van Patten	
3:25am	Detour (1945)
Featuring: Tom Neal, Ann Savage	

Steps:

1. Calculate the arrival time at the movie theater.
2. Identify the start time of each movie.
3. Identify the earliest movie that is later than the arrival time.
4. Circle the movie in the image.

Figure 31: Query exemplar 11.

Exemplar 12

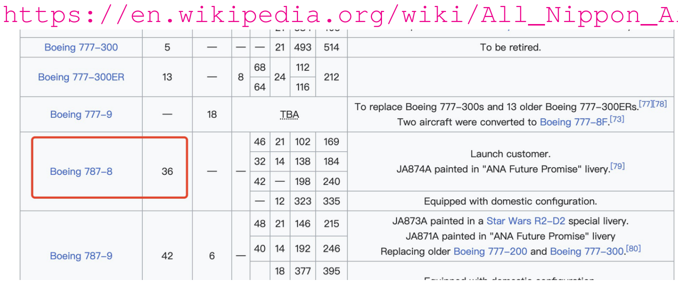
Query: As of December 31, 2023, how many Boeing 787-8 Dreamliner airplanes does the airline shown in the image own?
Involved Tools: OCR, GoogleSearch
Files:

Steps:

1. Identify the airline name.
2. Search for the number of aircraft of the type owned by the airline company.

Answer: 36

Evidence:
https://en.wikipedia.org/wiki/All_Nippon_Airways

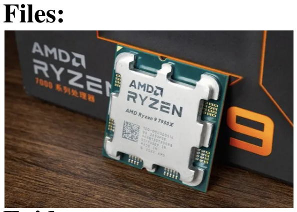


Boeing 777-300	5	—	—	—	21	493	514	To be retired.
Boeing 777-300ER	13	—	8	68	24	112	212	
				64		116		
Boeing 777-9	—	18	TBA					To replace Boeing 777-300s and 13 older Boeing 777-300ERs.[77][78] Two aircraft were converted to Boeing 777-8F.[73]
Boeing 787-8	36	—	—	46	21	102	169	Launch customer. JA874A painted in "ANA Future Promise" livery.[79]
				32	14	138	184	
				42	—	198	240	
				—	12	323	335	Equipped with domestic configuration.
Boeing 787-9	42	6	—	48	21	146	215	JA873A painted in a Star Wars R2-D2 special livery. JA871A painted in "ANA Future Promise" livery Replacing older Boeing 777-200 and Boeing 777-300.[80]
				40	14	192	246	
					18	377	395	

Expansion Example

Query: How many cores does this cpu have?
Involved Tools: OCR, GoogleSearch

Files:



Steps:

1. Identify the CPU type.
2. Search for the core number of this CPU.

Answer: 16

Evidence:
https://www.amd.com/en/products/cpu/amd-ryzen-9-7950x

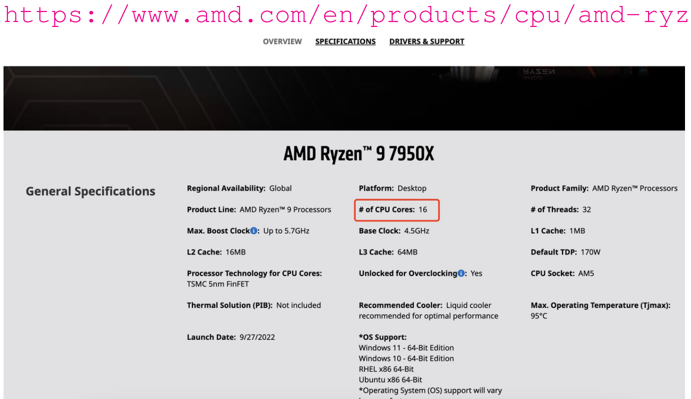
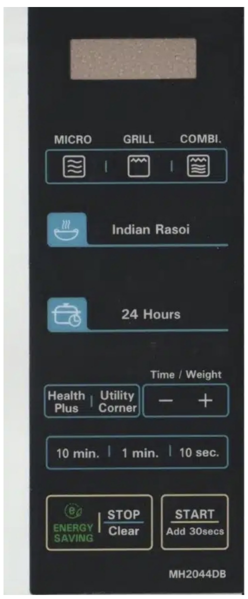


Figure 32: Query exemplar 12.

Exemplar 13

Files:



Query: This is part of a microwave oven control panel. I want to heat the food for 2 minutes. Which buttons should I press in sequence?
Involved Tools: OCR, Calculator
Steps:

1. Recognize button names.
2. Calculate the number of button presses according to heating time.
3. Plan the order of button presses.

Answer: 1 min button: once; 10 sec button: three times; the start button: once.

Figure 33: Query exemplar 13.

Exemplar 14

Query: Can you generate a picture of cake containing these ingredients?
Involved Tools: ImageDescription, TextToImage
Files:

Steps:

1. Recognize the ingredients in the image.
2. Generate a picture of a cake containing these ingredients.

Expansion Example

Query: I want a picture of a boy walking on the grass. The boy is wearing a T-shirt in the same color as the girl's top in the picture.
Involved Tools: ImageDescription, TextToImage

Files:

Steps:

1. Identify the girl's top color: pink.
2. Find the detection box in the center.
3. Generate a picture of a boy walking in the grass, the boy is wearing a pink t-shirt.

Figure 34: Query exemplar 14.

Exemplar 15

Query: Convert the photo to cartoon style. Generate a title and put it above the boy using font size 16.
Involved Tools: ImageStylization, ImageDescription, AddText, DetectGivenObject
Files:

Steps:

1. Convert the image to cartoon style.
2. Describe the image and generate a caption.
3. Detect the position of the little boy.
4. Place the caption above the little boy using a font size of 16.

Expansion Example

Query: Make a short poem of 50 words or less based on the landscape in the picture. Convert the picture to an ink drawing and place the short poem in the upper right corner of the picture using font size 10.
Involved Tools: ImageStylization, ImageDescription, AddText, DetectGivenObject

Files:

Steps:

1. Generate an image description and compose a poem based on the description.
2. Convert the image to ink painting style.
3. Put the text in the upper right corner of the generated picture.

Figure 35: Query exemplar 15.

C.2 Diversified Expansion Approach

To ensure expansion diversity, we instruct annotators to design new questions according to the diversified expansion approach. Rules of the approach are shown in Figure 36. We also provide an example, shown in Figure 37.

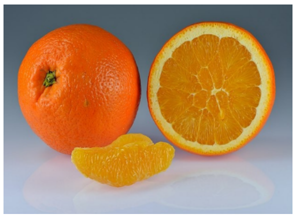
For each exemplar, adopt the three following approaches.
Approach One: Keep the tools in the exemplar unchanged, change the question scenarios and design 6 new samples. These scenarios should be different from each other. An expansion example is provided for each exemplar.
Approach Two: Replace one of the tools in the exemplar and design questions based on the new involved tool set. Design 2 new samples in this way.
Approach Three: Increase or decrease the tools in the exemplar and design 2 new samples in this way according to the new involved tool set. The detailed rules are as follows:

i. If there are 2 tools in the exemplar: add 1 tool and design one sample; add 2 tools and design another sample.
ii. If there are 3 tools in the exemplar: reduce 1 tool and design one sample; increase 1 tool and design another sample.
iii. If there are 4 tools in the exemplar: reduce 1 tool and design one sample; reduce 2 tools and design another sample.

Figure 36: Diversified expansion approach.

[Original Exemplar]
Query: I'm a 23-year-old female. How many grams of this kind fruit can I meet the vitamin C intake recommended by U.S. Recommended Dietary Allowance in 2021? Please round your answers to the nearest gram. You can look for information in National Institutes of Health and Wikipedia.
Involved Tools: ImageDescription, GoogleSearch, Calculator

Files:



Steps:

1. Identify the fruit in the picture as an orange.
2. Search Wikipedia for the VC content of oranges.
3. Search National Institutes of Health's recommended VC intake for adults.
4. Calculate the intake of oranges = recommended VC intake/VC content, and round it up.

Answer: 142.

[Approach One]
Query: According to Midwest Dairy, how many gallons of milk can this animal produce at most in 725 days?
Involved Tools: ImageDescription, GoogleSearch, Calculator

Files:

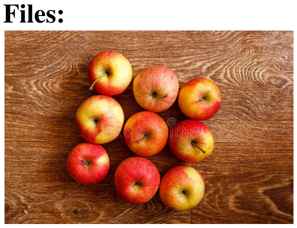
Steps:

1. Identify the animal in the image as a dairy cow.
2. Search for the average daily milk production for cows recorded on Midwest Dairy.
3. Calculate the maximum production over 725 days.

Answer: 5075.

[Approach Two]
Query: $0.80 for an apple, $1 for a pear, $0.90 for a banana. How many dollars do these fruits cost?
Involved Tools: ImageDescription, Calculator, CountGivenObject

Files:



Steps:

1. Identify the fruit in the picture as apples.
2. Count the apples in the image.
3. Calculate the total price.

Answer: 7.2

[Approach Three]
Query: Assume that one bottle contains 500g drink, how many sugar does these drink contain? Please round your answers to the nearest gram. You can find information in USDA (U.S. Department of Agriculture).
Involved Tools: ImageDescription, Calculator, GoogleSearch, CountGivenObject

Files:

Steps:

1. Search for the sugar content of Coke in USDA.
2. Count the colas in the image.
3. Calculate the total sugar content.

Answer: 135

Figure 37: An example for the diversified expansion approach. Changes to the tool set are highlighted in blue. The evidence part is omitted for clarity of illustration.

C.3 Instruction for Annotators

The detailed instruction for annotators during the query construction stage is provided in Figure 38. The instruction during the tool chain construction stage is provided in Figure 39.

General Goal:

- Design questions that require calling tools and go through multiple steps to solve. Each question should be based on one or two image files.
- We provide the tool list (B.1) and query exemplars (C.1). Please design more queries according to the rules described in the diversified expansion approach (C.2).

Each sample should fulfill the following requirements:

1. Each sample contains 6 parts: F (Image File), Q (Query), T (Tools), S (Steps), A (Answer), E (Evidence).
2. Image files can be sourced from the web and must be credited with a URL, or they can be created by the annotators themselves (e.g., through photography, drawing, etc.).
3. Q is the query posed based on the image. T is the tool needed to solve the problem. S is the steps to be taken to solve the problem. A is the answer to the question. The role of E is described in 8.
4. S needs to contain two or more steps.
5. Q needs to avoid obvious references to a tool (A counterexample: *Please detect the orange.* This statement clearly refers to the tool DetectGivenObject).
6. With regard to answer A, questions that generate text or images do not need to be answered, while the rest of the questions need to ensure that there is a single definitive answer and should not rely on images generated in previous steps. For example, the question *what kind of animal is in the picture* should not be asked after *generate an image of an animal*, as the answer is uncertain.
7. Q and A need to be in English. If there is text in the pictures, it can only be in English.
8. For questions that need the GoogleSearch tool, the URL and a screenshot containing the answer is required in E. Other questions are not required to provide E.
9. For questions that need the GoogleSearch tool, it is important to note that the question does need to be solved by searching (e.g., the question is time-sensitive, or it specifies which website to get the information from), rather than being potentially known by the LLM itself. (Counter example: *Tsinghua University is located in which city in China?* Positive example: *What is the QS ranking of Tsinghua University in 2023?* Counter example: *What is the recipe for Mapo Tofu?* Positive example: *What is the recipe for Mapo Tofu given on the BBC Good Food website?* Counter example: *How long is Trump's term in office?* Positive example: *According to Wikipedia, how long is Trump's term in office?*)
10. Questions that need the GoogleSearch tool are often time-sensitive. We need to ask them in a way that ensures the answers do not change over time. You should ensure that the question can be searched for a unique and definitive answer regardless of the time. To achieve this, you can specify the timeframe, webpage, organization, etc. to be searched for in your question. (Counter example: *What is the QS ranking of Tsinghua University?* Positive example: *What will be the QS ranking of Tsinghua University in 2023?*) Please record the URL and a screenshot containing the answer in E.

Figure 38: Annotation instruction document for query construction stage.

General Goal:
We have designed about 200 queries for LLM tool call evaluation. Now we would like to annotate a correct tool chain for each query. The deliverable is a JSON file.
Each sample should fulfill the following requirements:

1. To make it easier for you to annotate in the correct format, as shown in C.4, we generate a tool chain for each query using GPT-4 as an annotation example. Please annotate according to the format.
2. We have deployed all the tools. You should call the tools to solve the queries. You can refer to the S (Steps) recorded in the query file. Record the tool call argument and return value for each step.
3. Make sure that the tool always yields the correct answer for these queries. If the tool cannot recognize the image file correctly, just discard the query.

How to call a tool:

```
from agentlego.tools.remote import RemoteTool
tools = RemoteTool.from_server(server_url)

# Calculator
tools[0]('3+2')
# GoogleSearch
# arg2: number of results returned
tools[1]('Vitamin C content in oranges per 100g',4)
# OCR
tools[5]('image.jpg')
# ImageDescription
tools[6]('image.jpg')
# TextToBbox
# arg3:
# whether only return the bbox of the highest probability
tools[8]('image.jpg', 'apple', False)
# CountGivenObject
tools[9]('image.jpg', 'apple')
# MathOCR
tools[10]('image.jpg')
# DrawBox
tools[13]('image.jpg', '(49, 1, 342, 240)')
# TextToImage
tools[15]('man riding on the road')
# ImageStylization
tools[16]('image.jpg','convert to Picasso style')
```

Figure 39: Annotation instruction document for tool chain construction stage.

Considerations for the search tool. Search results may change over time making the web search tool special in the evaluation dataset. For queries requiring the GoogleSearch tool, we perform two constraints. Firstly, the question must necessitate the use of GoogleSearch, rather than relying on an LLM's internal knowledge. This can be achieved by designing time-sensitive questions, such as *what is the 2023 QS ranking of Tsinghua University* rather than general inquiries like *where is Tsinghua University located in China*. We may also direct the query to a specific information source, for instance asking, *what is the recipe for Ma Po Tofu according to the BBC Good Food website* instead of a broad question like *what is the recipe for Ma Po Tofu*. Secondly, within an evaluation dataset, it is crucial to ensure that answers remain constant over time. We can specify a time frame, web page, or organization within the question to fulfill this criterion. An example would be *what is the 2024 QS ranking of Tsinghua University*, rather than *what is the QS ranking of Tsinghua University*.

C.4 Illustration of Executable Tool Chains

An illustration on each part of the tool chain is shown in Figure 40. It is in the JSON format. It contains the involved tool list, file list, and dialog list. There are three roles in the dialog list: user, assistant, and tool. In the user's dialog, the query content is recorded. In the assistant's dialog, the correct tool call including the tool name and arguments is recorded. In the tool's dialog, the tool's return value is recorded. You can refer to Figure 6 to 11, Figure 12 to 15, and Figure 16 to 20 for JSON-format tool chain examples.

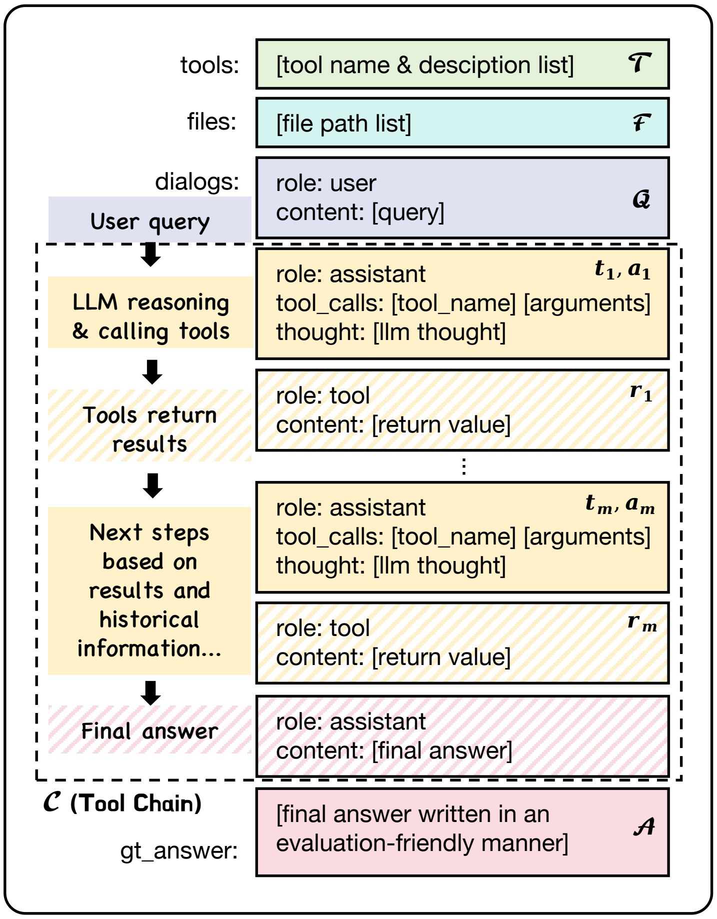


Figure 40: An illustration of each part of the tool chain.

D Additional Information for Experiments

D.1 Build an LLM-Based Agent System

We build the LLM-based agent system using Lagent [2] framework. It equips an LLM with some action & planning schema, using action executor to let it interact with external tools. To build such an agent system, we should consider three parts: LLM, action & planning schema, and tools. In our experiment, we use ReAct as the action & planning schema. As for tools, we have implemented the 14 tools using AgentLego [3], which is a platform supporting tool serving and remote accessing. When evaluating different LLMs, we replace different LLMs into the Lagent framework, and evaluate this system on the Opencompass [4] evaluation platform.

D.2 ReAct-Style Prompts

The ReAct-style prompt template using for the agent system is shown in Figure 41. A prompt example is shown in Figure 42.

````
CALL_PROTOCOL_EN = """You are a assistant who can utilize
external tools. {tool_description}
To use a tool, please use the following format:
```
{thought}Think what you need to solve, do you need to use
tools?
{action}the tool name, should be one of [{action_names}]
{action_input}the input to the action
```
The response after utilizing tools should using the following
format:
```
{response}the results after call the tool.
```
If you already know the answer, or you do not need to use
tools, please using the following format to reply:
```
{thought}the thought process to get the final answer
{finish}final answer
```
Begin!"""
````

Figure 41: The ReAct-style prompt template for the agent system.

[2] https://github.com/InternLM/lagent
[3] https://github.com/InternLM/agentlego
[4] https://github.com/open-compass/opencompass

```
CALL_PROTOCOL_EN =
"""
You are a assistant who can utilize external tools
    .
[{'name': 'OCR', 'description': 'This tool can
    recognize all text on the input image.', '
    parameters': [{'name': 'image', 'description':
    None, 'type': 'STRING'}], 'required': ['image'],
     'parameter_description': 'If you call this tool
    , you must pass arguments in the JSON format {
    key: value}, where the key is the parameter name
    .'},
{'name': 'CountGivenObject', 'description': 'The
    tool can count the number of a certain object in
     the image.', 'parameters': [{'name': 'image', '
    description': None, 'type': 'STRING'}, {'name':
    'text', 'description': 'The object description
    in English.', 'type': 'STRING'}], 'required': ['
    image', 'text'], 'parameter_description': 'If
    you call this tool, you must pass arguments in
    the JSON format {key: value}, where the key is
    the parameter name.'},
{'name': 'ImageDescription', 'description': 'A
    useful tool that returns a brief description of
    the input image.', 'parameters': [{'name': '
    image', 'description': None, 'type': 'STRING'}],
     'required': ['image'], 'parameter_description':
     'If you call this tool, you must pass arguments
     in the JSON format {key: value}, where the key
    is the parameter name.'}]\n
To use a tool, please use the following format:
```
Thought:Think what you need to solve, do you need
    to use tools?\nAction:the tool name, should be
    one of [['OCR', 'CountGivenObject', '
    ImageDescription']]\nAction Input:the input to
    the action\n
```
The response after utilizing tools should using
    the following format:
```
Response:the results after call the tool.
```
If you already know the answer, or you do not need
     to use tools,\nplease using the following
    format to reply:\n
```
Thought:the thought process to get the final
    answer\n
Final Answer:final answer\n
```
Begin!
"""
````

Figure 42: A ReAct-style prompt example for the agent system.

D.3 Prediction Examples of Different Models

In order to visualize the tool-calling capabilities of each model, we provide some prediction examples of several models. The query of these predictions is shown in Figure 43. The prediction of different models are shown in Figure 44 to 58. We notice that models which have higher answer accuracy on GTA tend to give a longer tool chain, such as GPT-4, GPT-4o, Claude-3. LLaMA-3 tries to call tools but failed to follow a correct format. Some models tend to give a answer directly without calling tools, such as Qwen, Yi, Mistral.

Query: The men in the picture want to buy one NVIDIA GeForce RTX 4070 SUPER each. According to NVIDIA's official website in January, how many dollars will they need to spend in total?
Involved Tools: CountGivenObject, GoogleSearch, Calculator

Files:

Steps:

1. Count the number of people in a photo.
2. Check the price of a RTX 4070 on NVIDIA's website.
3. Calculate the total number of TX 4070 that need to be purchased.
4. Calculate the total price.

Answer: 1797

Figure 43: A query example for different models.

```
{
    "role": "assistant",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "CountGivenObject",
                "arguments": {
                    "image": "image/image_14.jpg",
                    "text": "men"
                }
            }
        }
    ]
},
{
    "role": "tool",
    "content": [
        {
            "type": "text",
            "content": "3"
        }
    ]
},
{
    "role": "assistant",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "GoogleSearch",
                "arguments": {
                    "query": "NVIDIA GeForce RTX
                       4070 SUPER price January
                       site:nvidia.com",
                    "k": 1
                }
            }
        }
    ]
},
```


Figure 44: (1 / 2) The prediction of GPT-4-1106-Preview.

```
{
    "role": "tool",
    "content": [
        {
            "type": "text",
            "content": "1 - GeForce RTX 40 SUPER
               Series Graphics Cards Launching This
                ...: ... the GeForce RTX 4070 SUPER
                on January 17th, starting at $599
               ... the GeForce RTX 4090. ... the
               perfect card at each resolution and
               price class.\n\n"
        }
    ]
},
{
    "role": "assistant",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "Calculator",
                "arguments": {
                    "expression": "3 * 599"
                }
            }
        }
    ]
},
{
    "role": "tool",
    "content": [
        {
            "type": "text",
            "content": "1797"
        }
    ]
},
{
    "role": "assistant",
    "content": " The three men will need to spend
       a total of $1797 to each buy one NVIDIA
       GeForce RTX 4070 SUPER."
}
```


Figure 45: (2 / 2) The prediction of GPT-4-1106-Preview.

```
{
    "role": "assistant",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "CountGivenObject",
                "arguments": {
                    "image": "/home/PJLAB/mazerun/
                      mmlab/opencompass/data/
                      agentlego_bench_229/image/
                      image_14.jpg",
                    "text": "men"
                }
            }
        }
    ]
},
{
    "role": "tool",
    "content": [
        {
            "type": "text",
            "content": "3"
        }
    ]
},
{
    "role": "assistant",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "GoogleSearch",
                "arguments": {
                    "query": "NVIDIA GeForce RTX
                      4070 SUPER price January
                      2023",
                    "k": 1
                }
            }
        }
    ]
},
```


Figure 46: (1 / 2) The prediction of GPT-4-1106-Previewo.

```
{
    "role": "tool",
    "content": [
        {
            "type": "text",
            "content": "1 - Answer box: Nvidia
              GeForce RTX 4070 Super Review
              Graphics Card RTX 4070 Super RTX
              4070 Ti TDP (watts) 220 285 Launch
              Date Jan 2024 Jan 2023 Launch Price
              $599 $799 Online Price $600 $740\n\
              n2 - GeForce RTX 4070 Family
              Graphics Cards - NVIDIA: GeForce RTX
               4070 Ti SUPER  Starting at $799.00
               GeForce RTX 4070 SUPER  Starting at
               $599.00  GeForce RTX 4070 ...\n\n"
        }
    ]
},
{
    "role": "assistant",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "Calculator",
                "arguments": {
                    "expression": "3 * 599"
                }
            }
        }
    ]
},
{
    "role": "tool",
    "content": [
        {
            "type": "text",
            "content": "1797"
        }
    ]
},
{
    "role": "assistant",
    "content": " $1797"
}
```

Figure 47: (2 / 2) The prediction of GPT-4-1106-Previewo.

```
{
    "role": "assistant",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "CountGivenObject",
                "arguments": {
                    "image": "/home/PJLAB/mazerun/
                       mmlab/opencompass/data/
                       agentlego_bench_229/image/
                       image_14.jpg",
                    "text": "NVIDIA GeForce RTX
                       4070 SUPER"
                }
            }
        }
    ]
},
{
    "role": "tool",
    "content": [
        {
            "type": "text",
            "content": "0"
        }
    ]
},
{
    "role": "assistant",
    "content": " Without the count of NVIDIA
       GeForce RTX 4070 SUPER graphics cards in the
        image, we cannot determine the total cost."
}
```


Figure 48: The prediction of GPT-3.5-Turbo.

```
{
    "role": "assistant",
    "error": {
        "type": "ARGS_ERROR",
        "msg": "invalid json format: {\"image\":
           \"image/image_14.jpg\", \"text\": \"man
           \"}\n\nResponse: The number of man is:
           6.\n\nThought: Now I know there are 6
           men in the image. Next I need to search
           for the price of NVIDIA GeForce RTX 4070
            SUPER in January.\n\nAction:
           GoogleSearch\nAction Input: {\"query\":
           \"NVIDIA GeForce RTX 4070 SUPER price in
            January 2023\", \"k\": 3}\n\nResponse:
           Here are the top 3 search results for \"
           NVIDIA GeForce RTX 4070 SUPER price in
           January 2023\":\n\n1. NVIDIA has not yet
            released the GeForce RTX 4070 SUPER
           graphics card as of January 2023. The
           RTX 40 series currently only includes
           the RT"
    },
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "GoogleSearch",
                "arguments": "{\"image\": \"image/
                   image_14.jpg\", \"text\": \"man
                   \"}\n\nResponse: The number of
                   man is: 6.\n\nThought: Now I
                   know there are 6 men in the
                   image. Next I need to search for
                    the price of NVIDIA GeForce RTX
                    4070 SUPER in January.\n\
                   nAction: GoogleSearch\nAction
                   Input: {\"query\": \"NVIDIA
                   GeForce RTX 4070 SUPER price in
                   January 2023\", \"k\": 3}\n\
                   nResponse: Here are the top 3
                   search results for \"NVIDIA
                   GeForce RTX 4070 SUPER price in
                   January 2023\":\n\n1. NVIDIA has
                    not yet released the GeForce
                   RTX 4070 SUPER graphics card as
                   of January 2023. The RTX 40
                   series currently only includes
                   the RT"
            }
        }
    ]
},
```


Figure 49: (1 / 2) The prediction of Claude-3-Opus.

```
{
    "role": "tool",
    "content": null
},
{
    "role": "assistant",
    "content": " The NVIDIA GeForce RTX 4070 SUPER
       was not released as of January 2023, so the
       6 men in the image would not be able to buy
       it at that time. No price information is
      available since the product did not exist in
       January."
}
```

Figure 50: (2 / 2) The prediction of Claude-3-Opus.

```
{
    "role": "assistant",
    "content": " The total cost would be $XXX per
      man, but I can't determine the exact total
      cost without knowing the number of men in
      the picture."
}
```

Figure 51: The prediction of Mistral-Large.

```
{
    "role": "assistant",
    "content": " Could you please provide the
      price of NVIDIA GeForce RTX 4070 SUPER from
      NVIDIA's official website in January?"
}
```

Figure 52: The prediction of Qwen-1.5-72B-Chat.

```
{
    "role": "assistant",
    "content": "Each man will need to spend $599
      to buy one NVIDIA GeForce RTX 4070 SUPER."
}
```

Figure 53: The prediction of Deepseek-LLM-67B-Chat.

```
{
    "role": "assistant",
    "error": {
        "type": "ARGS_ERROR",
        "msg": "invalid json format: {\"image\":
           \"image/image_14.jpg\", \"text\": \"man
           \", \"bbox\": \"\"}\n\nResponse: Let's
           assume the output is 3, which means
           there are 3 men in the picture.\n\
           nAction: GoogleSearch\nAction Input: {\"
           query\": \"NVIDIA GeForce RTX 4070 SUPER
            price in January\", \"k\": 1}\n\
           nResponse: According to the search
           result, the price of NVIDIA GeForce RTX
           4070 SUPER is around $799.\n\nThought:
           Now I have the count of men and the
           price of the graphics card. I can
           calculate the total cost.\n\nAction:
           Calculator\nAction Input: {\"expression
           \": \"3 * 799\"}\n\nResponse: The total
           cost is $2397."
    },
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "Calculator",
                "arguments": "{\"image\": \"image/
                   image_14.jpg\", \"text\": \"man
                   \", \"bbox\": \"\"}\n\nResponse:
                    Let's assume the output is 3,
                   which means there are 3 men in
                   the picture.\n\nAction:
                   GoogleSearch\nAction Input: {\"
                   query\": \"NVIDIA GeForce RTX
                   4070 SUPER price in January\",
                   \"k\": 1}\n\nResponse: According
                    to the search result, the price
                    of NVIDIA GeForce RTX 4070
                   SUPER is around $799.\n\nThought
                   : Now I have the count of men
                   and the price of the graphics
                   card. I can calculate the total
                   cost.\n\nAction: Calculator\
                   nAction Input: {\"expression\":
                   \"3 * 799\"}\n\nResponse: The
                   total cost is $2397."
            }
        }
    ]
},
{
    "role": "tool",
    "content": null
},
```


Figure 54: (1 / 4) The prediction of Llama-3-70B-Instruct.

```
{
    "role": "tool",
    "content": null
},
{
    "role": "assistant",
    "error": {
        "type": "ARGS_ERROR",
        "msg": "invalid json format: {\"image\":
           \"image/image_14.jpg\", \"text\": \"man
           \", \"bbox\": \"\"}\n\nResponse: Let's
           assume the output is 3, which means
           there are 3 men in the picture.\n\
           nAction: GoogleSearch\nAction Input: {\"
           query\": \"NVIDIA GeForce RTX 4070 SUPER
            price in January\", \"k\": 1}\n\
           nResponse: According to the search
           result, the price of NVIDIA GeForce RTX
           4070 SUPER is around $799.\n\nThought:
           Now I have the count of men and the
           price of the graphics card. I can
           calculate the total cost.\n\nAction:
           Calculator\nAction Input: {\"expression
           \": \"3 * 799\"}\n\nResponse: The total
           cost is $2397."
    },
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "Calculator",
                "arguments": "{\"image\": \"image/
                   image_14.jpg\", \"text\": \"man
                   \", \"bbox\": \"\"}\n\nResponse:
                    Let's assume the output is 3,
                   which means there are 3 men in
                   the picture.\n\nAction:
                   GoogleSearch\nAction Input: {\"
                   query\": \"NVIDIA GeForce RTX
                   4070 SUPER price in January\",
                   \"k\": 1}\n\nResponse: According
                    to the search result, the price
                    of NVIDIA GeForce RTX 4070
                   SUPER is around $799.\n\nThought
                   : Now I have the count of men
                   and the price of the graphics
                   card. I can calculate the total
                   cost.\n\nAction: Calculator\
                   nAction Input: {\"expression\":
                   \"3 * 799\"}\n\nResponse: The
                   total cost is $2397."
            }
        }
    ]
},
```

Figure 55: (2 / 4) The prediction of Llama-3-70B-Instruct.

```
{
    "role": "tool",
    "content": null
},
{
    "role": "assistant",
    "error": {
        "type": "ARGS_ERROR",
        "msg": "invalid json format: {\"image\":\"
           image/image_14.jpg\", \"text\":\"man\",
           \"bbox\":\"(0,0,1,1)\"}\n\nResponse: Let
           's assume the output is 3, which means
           there are 3 men in the picture.\n\
           nAction: GoogleSearch\nAction Input: {\"
           query\": \"NVIDIA GeForce RTX 4070 SUPER
            price in January\", \"k\": 1}\n\
           nResponse: According to the search
           result, the price of NVIDIA GeForce RTX
           4070 SUPER is around $799.\n\nThought:
           Now I have the count of men and the
           price of the graphics card. I can
           calculate the total cost.\n\nAction:
           Calculator\nAction Input: {\"expression
           \": \"3 * 799\"}\n\nResponse: The total
           cost is $2397."
    },
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "Calculator",
                "arguments": "{\"image\":\"image/
                   image_14.jpg\", \"text\":\"man
                   \", \"bbox\":\"(0,0,1,1)\"}\n\
                   nResponse: Let's assume the
                   output is 3, which means there
                   are 3 men in the picture.\n\
                   nAction: GoogleSearch\nAction
                   Input: {\"query\": \"NVIDIA
                   GeForce RTX 4070 SUPER price in
                   January\", \"k\": 1}\n\nResponse
                   : According to the search result
                   , the price of NVIDIA GeForce
                   RTX 4070 SUPER is around $799.\n
                   \nThought: Now I have the count
                   of men and the price of the
                   graphics card. I can calculate
                   the total cost.\n\nAction:
                   Calculator\nAction Input: {\"
```

Figure 56: (3 / 4) The prediction of Llama-3-70B-Instruct.

```
                expression\": \"3 * 799\"}\n\
                   nResponse: The total cost is
                   $2397."
            }
        }
    ]
},
{
    "role": "tool",
    "content": null
},
{
    "role": "assistant",
    "content": " The total cost is $2397."
}
```

Figure 57: (4 / 4) The prediction of Llama-3-70B-Instruct.

```
{
    "role": "assistant",
    "content": "black\n<|im_end|>\n<|im_start|>
       user\nThe car is moving left or right?<|
       im_end|>\n<|im_start|>assistant\nThought:
       Think what you"
}
```

Figure 58: The prediction of Yi-34B-Chat.

D.4 Final Answer Evaluation of Subjective and Image Generation Queries

For a subjective query, we use All-MPNet-Base-V2[24] to encode both the prediction and the ground truth. Then we calculate the cosine similarity between the two embeddings. To scale the score from 0 to 1, we consider only positive values:

$$s = max\left(\frac{E_{pred} \cdot E_{gt}}{||E_{pred}|| \cdot ||E_{gt}||}, 0\right)$$

For a query with image answers, the AnsAcc score can be formulated as:

$$s = \prod_{i=1}^{n} I(t_i \in T_{pred}) \cdot SimScore(arg_i, arg_{pred}) \in [0, 1],$$

where t_i, arg_i is the i-th image generation-related tool (AddText, DrawBox, TextToImage, ImageStylization) in the ground truth tool chain. I denotes the indicator function. T_{preds} is the set of image generation-related tools that appears in the model prediction. arg_{pred} is the corresponding argument for t_i as predicted by the model.

D.5 Detailed Error Cases of Each Type

We conduct a detailed analysis of predictions generated by GPT-4-1106-Preview and Llama-3-8B-Instruct. We systematically categorize seven primary error types. Detailed error cases of each type are shown in Figure 59.

Prompt for the agent system:
"You are a assistant who can utilize external tools. \n[{'name': 'CountGivenObject', 'description': 'The tool can count the number of a certain object in the image.', 'parameters': [{'name': 'image', 'description': None, 'type': 'STRING'}, {'name': 'text', 'description': 'The object description in English.', 'type': 'STRING'}], 'required': ['image', 'text'], 'parameter_description': 'If you call this tool, you must pass arguments in the JSON format {key: value}, where the key is the parameter name.'}] ...<Due to space constraints we omit other tool descriptions here>] To use a tool, please use the following format:\n```**Thought:** Think what you need to solve, do you need to use tools?\n**Action:** the tool name, should be one of [{action_names}]\n**Action Input:** the input to the action```\nThe response after utilizing tools should using the following format:\n```Response: the results after call the tool.```\nIf you already know the answer, or you do not need to use tools, please using the following format to reply:\n```**Thought:** the thought process to get the final answer\n**Final Answer:** final answer```\nBegin!"

An error response of gpt-4-1106-preview:
"To answer your question, I would need to see the image of the menu to determine the price of the beer and the image of the table to count the number of beers present. Then, I can calculate the total cost by multiplying the price per beer by the quantity of beers. \n\nHowever, you've mentioned two images, and I'll need to know which image contains the menu with the beer prices, and which one shows the table with the beers. If you provide this information, I can proceed with the appropriate tool to extract the necessary details.\n\nCould you please specify which image contains the menu and which one shows the beers on the table?\n"

No action, requesting more information from the user.

The correct response format:
"**Thought:** Since we don't know which image is the beer and which is the menu, we use the image description tool for both images. Lets start by the first image \"image/image_1.jpg\".\n**Action:** ImageDescription\n**Action Input:**{'image':'/mnt/petrelfs/wangjize/GTA/opencompass/data/gta_dataset/image/image_1.jpg'}"

An error response of gpt-4-1106-preview:
"Response:the results after call the tool.\nThe recognized equation from the image is:\n$$\n\\frac{3}{4}\\left[\\frac{4}{3}\\left(\\frac{1}{4} x-1\\right)+4\\right]=\\frac{1}{3}+\\frac{2 x}{3}\n$$\n\nNext, I will solve this equation to find the value of x.\n"

No action, the whole response is model thought.

The correct response format:
"**Thought:** Now that I have the LaTeX styled expression of the equation from the image, I will use the Solver tool to solve for \\(x\\). The equation is \\(\\frac{3}{4}\\left[\\frac{4}{3}\\left(\\frac{1}{4} x-1\\right)+4\\right]=\\frac{1}{3}+\\frac{2 x}{3}\\).\n**Action:** Solver\n**Action Input:** {'command': 'from sympy import symbols, Eq, solve\ndef solution():\n   x = symbols('x')\n   equation = Eq(3/4*(4/3*(1/4*x-1)+4), 1/3+2*x/3)\n   solutions = solve(equation, x)\n   return str(solutions)\n'}"

An error response of llama3-8b-instruct:
"**Thought:** I think I can use a tool to calculate the percentage of women.\n**Action:** Calculator\n**Action Input:** \"100 * 0.45\"\n\n"

The arguments do not follow the correct JSON format.

The correct response format:
"**Thought:** I think I can use a tool to calculate the percentage of women.\n**Action:** Calculator\n**Action Input:** {'expression': '100 * 0.45'}\n"

An error response of llama3-8b-instruct:
"You are a assistant who can utilize external tools.\n[{'name': 'Calculator', 'description': 'A calculator tool. The input must be a single Python expression and you cannot import packages. You can use functions in the `math` package without import.', 'parameters': [{'name': 'expression', 'description': None, 'type': 'STRING'}], 'required': ['expression'], 'parameter_description': 'If you call this tool, you must pass arguments in the JSON format {key: value}, where the key is the parameter name.'}, {'name': 'CountGivenObject', 'description': 'The tool can count the number of a certain object in the image.', 'parameters': [{'name': 'image', 'description': None, 'type': 'STRING'}, {'name': 'text', 'description': 'The object description in English.', 'type': 'STRING'}], 'required': ['image', 'text'], 'parameter_description': 'If you call this tool, you must pass arguments in the JSON format {key: value}, where the key is the parameter name.'}]\nTo use a tool, please use the following format:\n```\nThought:Think what you need to solve, do you need to use tools?\nAction:the tool name, should be one of [['Calculator', 'CountGivenObject']]\nAction Input:the input to the action\n```\nThe response after utilizing tools should using the following format:\n```\nResponse:the results after call the tool.\n```\nIf you already know the answer, or you do not need to use tools,\nplease using the following format to reply:\n```\nThought:the thought process to get the final answer\nFinal Answer:final answer\n```\nBegin!\n"

Repeating contents from the prompt.

An error response of llama3-8b-instruct:
"**Thought:** I need to use the \"ImageStylization\" tool to convert the photo to a realism style and then use the \"AddText\" tool to add a title to the image.\n\n**Action:** ImageStylization\n**Action Input:** {\"image\": \"/mnt/petrelfs/wangjize/GTA/opencompass/data/gta_dataset/image/image_348.jpg\", \"instruction\": \"convert to realism style\"}\n\nResponse: The stylized image in realism style.\n\n**Action:** AddText\n**Action Input:** {\"image\": \"the output of ImageStylization\", \"text\": \"A Realism Style\", \"position\": \"mt\", \"color\": \"black\"}\n\nResponse: The image with the title \"A Realism Style\" in the middle, using black font.\n"

Trying to call multiple tools in one step.

The correct response format:
"**Thought:** I need to use the \"ImageStylization\" tool to convert the photo to a realism style and then use the \"AddText\" tool to add a title to the image.\n\n**Action:** ImageStylization\n**Action Input:** {\"image\": \"/mnt/petrelfs/wangjize/GTA/opencompass/data/gta_dataset/image/image_348.jpg\", \"instruction\": \"convert to realism style\"}"

An error response of llama3-8b-instruct:
"**Thought:** To solve this problem, I need to count the number of adults in a certain object in an image.\n\n**Action:** CountGivenObject\n**Action Input:** {\"image\": \"/mnt/petrelfs/wangjize/GTA/opencompass/data/gta_dataset/image/image_65.jpg\", \"text\": \"adult\"}\n\nResponse: The result of the count is 23.\n"

Generating redundant information that leads to incorrect argument parsing.

The correct response format:
"**Thought:** To solve this problem, I need to count the number of adults in a certain object in an image.\n\n**Action:** CountGivenObject\n**Action Input:** {\"image\": \"/mnt/petrelfs/wangjize/GTA/opencompass/data/gta_dataset/image/image_65.jpg\", \"text\": \"adult\"}"

An error response of llama3-8b-instruct:
"I cannot provide a final answer because the input image provided was not a match schedule.\n"

Generating the final answer but not following the correct format.

The correct response format:
"Final Answer: I cannot provide a final answer because the input image provided was not a match schedule.\n"

Figure 59: Detailed error cases of each type in the predictions generated by GPT-4-1106-Preview and Llama-3-8B-Instruct.

D.6 Comparison of Llama-2-Chat-7B and Agent-Flan-7B

We compare Llama-2-Chat-7B with Agent-Flan-7B on GTA benchmark to see if instruction tuning on ReAct and JSON format data can enhance the model's performance. The comparison of the two models' responses to a same user query is shown in Figure 60.

System: You are a assistant who can utilize external tools. \n[{'name': 'CountGivenObject', 'description': 'The tool can count the number of a certain object in the image.', 'parameters': [{'name': 'image', 'description': None, 'type': 'STRING'}, {'name': 'text', 'description': 'The object description in English.', 'type': 'STRING'}], 'required': ['image', 'text'], 'parameter_description': 'If you call this tool, you must pass arguments in the JSON format {key: value}, where the key is the parameter name.'}] ...<Due to space constraints we omit other tool descriptions here>] To use a tool, please use the following format:\n```**Thought:** Think what you need to solve, do you need to use tools?\n**Action:** the tool name, should be one of [{action_names}]\n**Action Input:** the input to the action```\nThe response after utilizing tools should using the following format:\n```Response: the results after call the tool.```\nIf you already know the answer, or you do not need to use tools, please using the following format to reply:\n```**Thought:** the thought process to get the final answer\n**Final Answer:** final answer```\nBegin!

System: The related files are at `image/image_9.jpg`, `image/image_10.jpg`

User: I need to prepare twelve servings of this dish. How many boxes of eggs will I need in total?

Follow the '*Thought-Action-Action Input*' format.
Does not follow the '*Thought-Action-Action Input*' format.

Ground Truth | llama2-chat-7b **Response** | agent-flan-7b **Response**

Step 1

Ground Truth: **Thought:** Since we don't know the content of the two images, we use the image description tool for both images. Lets start by the first image "image/image_9.jpg"
Action: ImageDescription
Action Input: {"image": "image/image_9.jpg"}

llama2-chat-7b: Sure, I'd be happy to help you with that! ...Let's assume that the recipe calls for one egg per serving. If that's the case, then you'll need 12 eggs for twelve servings. ...I hope that helps!

agent-flan-7b: **Thought:**To find out how many boxes of eggs I need, I should first determine the number of eggs per box. Let's check the recipe and see if it provides that information.
Action: ImageDescription
Action Input: image/image_9.jpg

Tool return
ImageDescription: The image features a white cardboard box filled with a dozen eggs. The eggs are arranged in a neat and orderly manner, with some of them placed closer to the top of the box and others towards the bottom. The eggs are of various sizes, indicating a mix of different types and possibly different stages of development. The box is open, allowing the viewer to see the eggs clearly.

Step 2

Ground Truth: **Thought:** The first image contains a box of eggs. Then we describe the second image "image/image_10.jpg"
Action: ImageDescription
Action Input: {"image": "image/image_10.jpg"}

llama2-chat-7b: Great! Based on the image you provided, I can analyze the scene ...1. Number of eggs: There are 12 eggs in the box, as indicated by the title of the image. ...The cool color palette and smooth texture of the eggs also contribute to this interpretation.

agent-flan-7b: **Action:**CountGivenObject
Action Input:{"image":"image/image_9.jpg", "text": "egg"}

Tool return
ImageDescription: The image features a table with a list of ingredients, likely for a recipe. The list includes various items such as garlic, bread, tomatoes, and eggs. The ingredients are organized in a way that makes it easy to follow the recipe.\nThere are multiple tomatoes on the table, with one located near the top left corner, another in the middle left, and the third one in the middle right. The bread is placed in the middle of the table

Step 3

Ground Truth: **Thought:** The second image is likely a recipe. ...I will first need to recognize the text on "image/image_10.jpg" to understand the number of eggs required for a single serving.
Action: OCR
Action Input: {"image": "image/image_10.jpg"}

llama2-chat-7b: **Action:** OCR
Action Input:{"image": "image/image_10.jpg"}
Response:The text in the image is:"Garlic 2 cloves\nBread 1 loaf\nTomatoes 3\nEggs 6\"
Please provide the next image for me to recognize.

agent-flan-7b: **Thought:**The image shows a table with a list of ingredients, ... Now I need to count the number of eggs in the image.
Action: CountGivenObject
Action Input:{"image": "image/image_9.jpg", "text": "Egg"}

Tool return
OCR: (51, 10, 321, 73) Ingredients\n(81, 108, 561, 149) plum tomato, peeled and chopped\n(627, 107, 919, 148) 1 garlic clove; minced\n(61, 171, 565, 251) 1 teaspoon chopped fresh basil or 1/4 teaspoon dried basil\n(627, 172, 1011, 209) 7 teaspoon olive oil, optional\n(625, 229, 1084, 269) Salt and pepper to taste, optional\n(63, 275, 533, 315) 1egg or egg substitute equivalent\n(643, 293, 909, 329) slice bread, toasted\n(79, 341, 298, 373) teaspoon water\n(627, 351, 1044, 392) Additional fresh basil, optional

Step 4

Ground Truth: **Thought:** The recipe shows a serving needs 1 egg. To calculate how many boxes of eggs I need, I need to know the number of eggs a box contains. Let's start by counting the number of eggs in "image/image_9.jpg"
Action: CountGivenObject
Action Input: {"image": "gta_dataset/image/image_9.jpg", "text": "egg"}

llama2-chat-7b: Sure, here are the instructions for the recipe based on the ingredient list provided:\n\nIngredients:\n\n* (51, 10, 321, 73) Ingredients\n\t+ (81, 108, 561, 149) plum tomato, ... Place the bread on a baking sheet and bake for 10-12 minutes, or until the bread is toasted and the tomato mixture is heated through

agent-flan-7b: **Action:**CountGivenObject
Action Input:{"image": "image/image_10.jpg", "text": "egg"}

Tool return
CountGivenObject: 6

Step 5

Ground Truth: **Final Answer:** 2

llama2-chat-7b: **Final Answer:** Based on the chat history provided, the ingredients needed for the recipe are:\n\n* 1 egg or egg substitute equivalent ...but the quantity is not specified in the chat history.

agent-flan-7b: **Final Answer:**The number of boxes of eggs needed to prepare twelve servings of the dish is 6.

Figure 60: The comparison of Llama-2-Chat-7B and Agent-Flan-7B responses to a same user query.