ToolACE: Wining the Points of LLM Function Calling

Weiwen Liu*, Xu Huang*, Xingshan Zeng*, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu°, Xinzhi Wang, Yong Liu, Yasheng Wang°, Duyu Tang, Dandan Tu, Lifeng Shang, Xin Jiang, Ruiming Tang°, Defu Lian°, Qun Liu, Enhong Chen

wwliu@sjtu.edu.cn; zeng.xingshan@huawei.com; xuhuangcs@mail.ustc.edu.cn

Motivation

Function calling significantly enhances the capability of AI Agents

- Access up-to-date information.
- Perform complex computations.
- Utilize third-party services.

Current tool-augmented LLMs primarily focus on simple function calling with limited diversity and complexity

- Single-turn function calling: Restrict dependent or multi-turn interactions.
- Limited generalization ability to new APIs: Reduce adaptability and scalability.

Table 1: Comparison of ToolACE with other representative tool-augmented LLMs (n/a represents not available.). ToolACE comprehensively incorporates the broadest range of APIs and domains, supports complex nested parameters (Nested), accommodates both parallel (Parallel) and dependent (Dependent) function calls, and addresses various types of tool-related data (Multi-type).

Model	#API	#Domain	Nested	Parallel	Dependent	Multi-type
Gorilla Patil et al. (2023)	1645	3	✗	✗	✗	✗
ToolAlpaca Tang et al. (2023)	3938	50	✗	✗	✗	✗
ToolLLM Qin et al. (2023)	16464	49	✗	✗	✓	✗
Functionary Meetkai (2024)	n/a	n/a	✗	✓	✗	✗
xLAM Liu et al. (2024)	3673	21	✗	✓	✗	✗
Granite Abdelaziz et al. (2024)	n/a	n/a	✗	✓	✗	✓
ToolACE	**26507**	**390**	✓	✓	✓	✓

ToolACE Data Generation Pipeline

- Evolutionary Diversity: Expose LLMs to a wide range of APIs and scenarios.
- Self-Guided Complexity: Generate sufficiently complex dialogues that align with the model's capabilities.
- Refined Accuracy: Ensure data quality through a combination of rule-based and model-driven Verification.

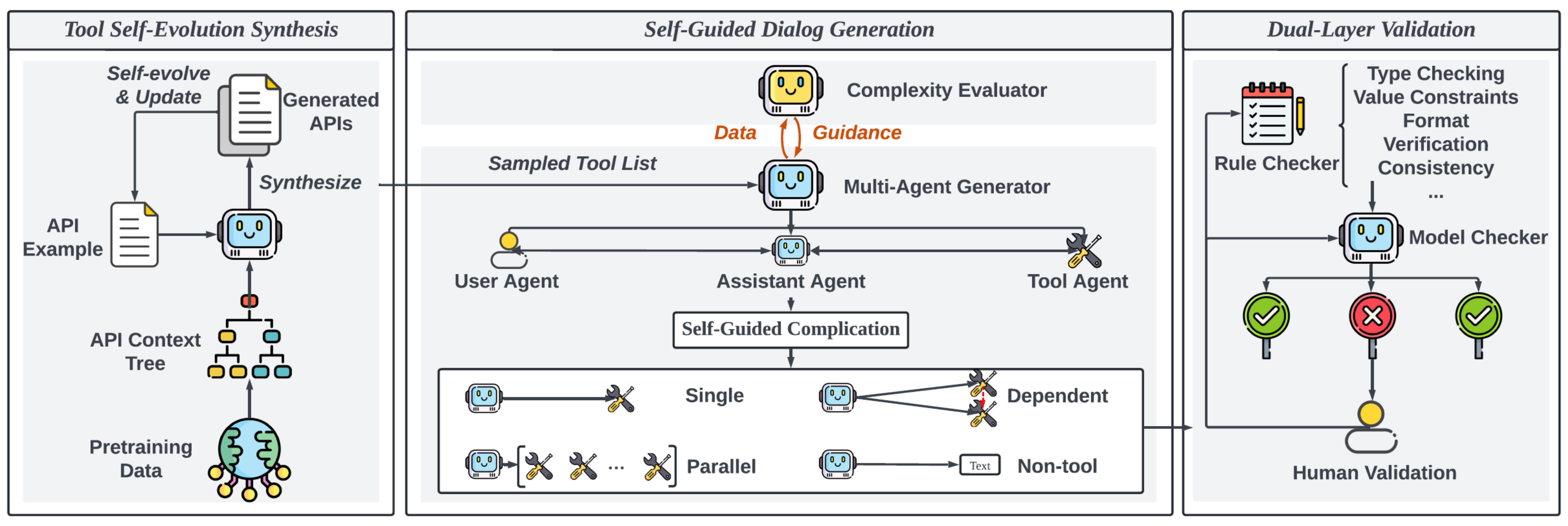

Figure 1: The overall framework of ToolACE, which mainly consists of Tool Self-evolution Synthesis (TSS), Self-Guided Dialog Generation (SDG), and Dual-Layer Validation Process (DLV).

Tool Self-Evolution Synthesis

- Speciation: Extract API functionalities from pretraining data and construct a context tree.
- Adaptation: Sample subtrees to assign unique functionalities to each API.
- Evolution: Apply diversity indicators (adding new functionalities or parameters) to generate new APIs.

Self-Guided Dialog Generation

- Multi-Agent Dialog Generation:
  - Simulate interactions among user, assistant, and tool agents.
  - Generate four types of function calling dialogs.
- Data Complexity Evaluation
  - Self-evaluate the data complexity and adjust the generation process accordingly

$$H_{\mathcal{M}}(x, y) = -\frac{1}{n_y}\sum_{i=1}^{n_y}\log p(t_i|x, t_1, \ldots, t_{i-1})$$

Dual-Layer Data Verification

- Rule Verification Layer: Ensure the data strictly adheres to the predefined syntactic and structural requirements
- Model Verification Layer: Incorporate LLMs to filter out data with hallucinations or inconsistencies.

Experiments

- We fine-tune the open-source LLaMA-3.1-8B-Instruct with ToolACE data, referred to as ToolACE-8B.
- ToolACE achieves state-of-the-art performance, comparable to the latest GPT-4 models.

Table 2: Accuracy performance comparison on BFCL-v3 leaderboard (updated on 09/20/2024). The top 20 models are listed for comparison. FC denotes the model is tailored for functional calling. (A) and (E) present AST and executable category, respectively. **Rel** and **Irrel** are abbreviations for relevance and irrelevance.

Rank	Overall	Model	Single turn Non-live (A)	Single turn Non-live (E)	Live (A)	Multi turn Multi turn	Hallucination Rel	Hallucination Irrel
1	59.49	GPT-4-turbo-2024-04-09 (FC)	82.65	83.80	73.39	21.62	70.73	79.79
2	59.29	GPT-4o-2024-08-06 (FC)	85.52	82.96	71.79	21.25	63.41	82.91
3	59.22	ToolACE-8B (FC)	89.27	90.07	73.21	14.37	85.37	83.81
4	59.13	xLAM-8x22b-r (FC)	89.75	89.32	72.81	15.62	97.56	75.23
5	58.45	GPT-4o-mini-2024-07-18 (FC)	82.83	81.80	67.53	25.75	82.93	71.83
6	57.94	xLAM-8x7b-r (FC)	88.44	85.89	71.97	15.75	92.68	72.35
7	57.21	GPT-4o-mini-2024-07-18 (Prompt)	86.54	87.95	72.77	11.62	80.49	79.20
8	55.82	mistral-large-2407 (FC)	84.12	83.09	67.17	20.50	78.05	48.93
9	55.67	GPT-4-turbo-2024-04-09 (Prompt)	91.31	88.12	67.97	10.62	82.93	61.82
10	54.83	Claude-3.5-Sonnet-20240620 (FC)	70.35	66.34	71.39	23.50	63.41	75.91
11	53.66	GPT-4o-2024-08-06 (Prompt)	80.90	77.89	73.88	6.12	53.66	89.56
12	53.43	GPT-4o1-mini-2024-09-12 (Prompt)	75.48	76.86	71.17	11.00	46.34	88.07
13	53.01	Gemini-1.5-Flash-Preview-0514 (FC)	77.10	71.23	71.17	13.12	60.98	76.15
14	52.53	Gemini-1.5-Pro-Preview-0514 (FC)	75.54	77.46	69.26	10.87	60.98	80.56
15	51.93	GPT-3.5-Turbo-0125 (FC)	84.52	81.66	59.00	19.12	97.56	35.83
16	51.78	FireFunction-v2 (FC)	85.71	84.23	61.71	11.62	87.80	52.94
17	51.78	Open-Mistral-Nemo-2407 (FC)	80.98	81.46	61.44	14.25	65.85	59.14
18	51.45	xLAM-7b-fc-r (FC)	86.83	85.02	68.81	0.00	80.49	79.76
19	51.01	Gorilla-OpenFunctions-v2 (FC)	87.29	84.96	68.59	0.00	85.37	73.13
20	49.63	Claude-3-Opus-20240229 (FC)	58.40	63.16	70.50	15.62	73.17	76.40
21	49.55	Meta-Llama-3-70B-Instruct (Prompt)	87.21	87.41	63.39	1.12	92.68	50.63

Table 3: Accuracy performance comparison on API-Bank evaluation system. **Bold** values represent the highest performance for API-based and open-source models, respectively.

	Model	Call	Retrieval+Call
API-based	gpt-3.5-turbo-0125	70.43	**52.59**
	gpt-4-0613	75.94	48.89
	gpt-4-turbo-2024-04-09	72.43	39.26
	gpt-4o-mini-2024-07-18	74.69	45.93
	gpt-4o-2024-05-13	**76.19**	42.96
Open-source	Alpaca-7B	24.06	5.19
	ChatGLM-6B	23.62	13.33
	Lynx-7B	49.87	30.37
	xLAM-7b-fc-r	32.83	21.48
	LLaMA-3.1-8B-Instruct	71.18	37.04
	ToolACE-8B	**75.94**	**47.41**

- In-Depth Analysis
  - Study on various backbone LLMs: Fine-tuning with ToolACE data yields substantial performance gains.
  - Study on general capabilities: Compared to the raw LLaMA-3.1-8B-Instruct, ToolACE-8B demonstrates negligible performance degradation on general benchmarks while achieving significant enhancements in function calling

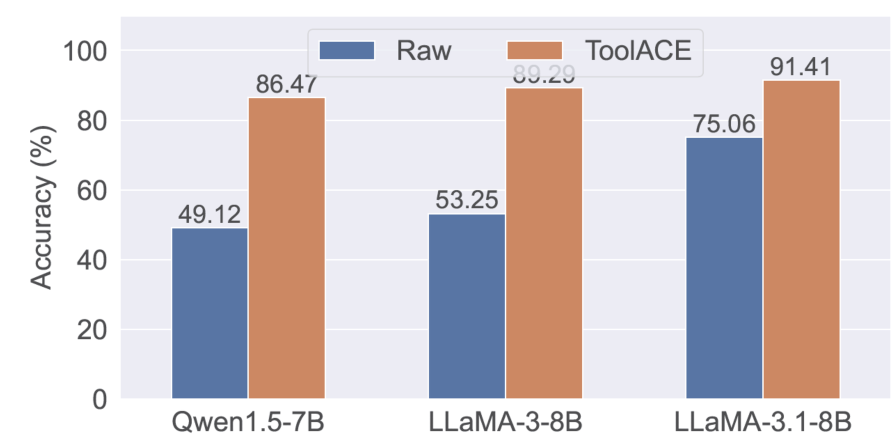

Figure 7: Performance on various LLMs.

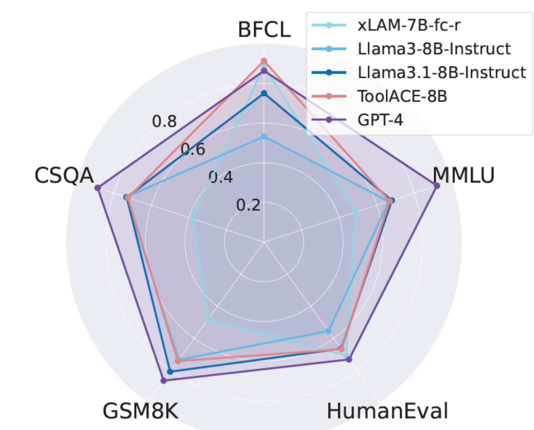

Figure 8: General capabilities.

Conclusion

- We propose a novel automated data pipeline for function calls, ToolACE. To our knowledge, this is the first work to highlight the benefits of synthesizing diverse APIs to improve the generalization of function calls.
- We develop a self-guided complication strategy to generate various types of function-calling dialogs with appropriate complexity.
- ToolACE significantly outperforms existing open-source LLMs and is competitive with the latest GPT-4 models.

Team-ACE