Our Work

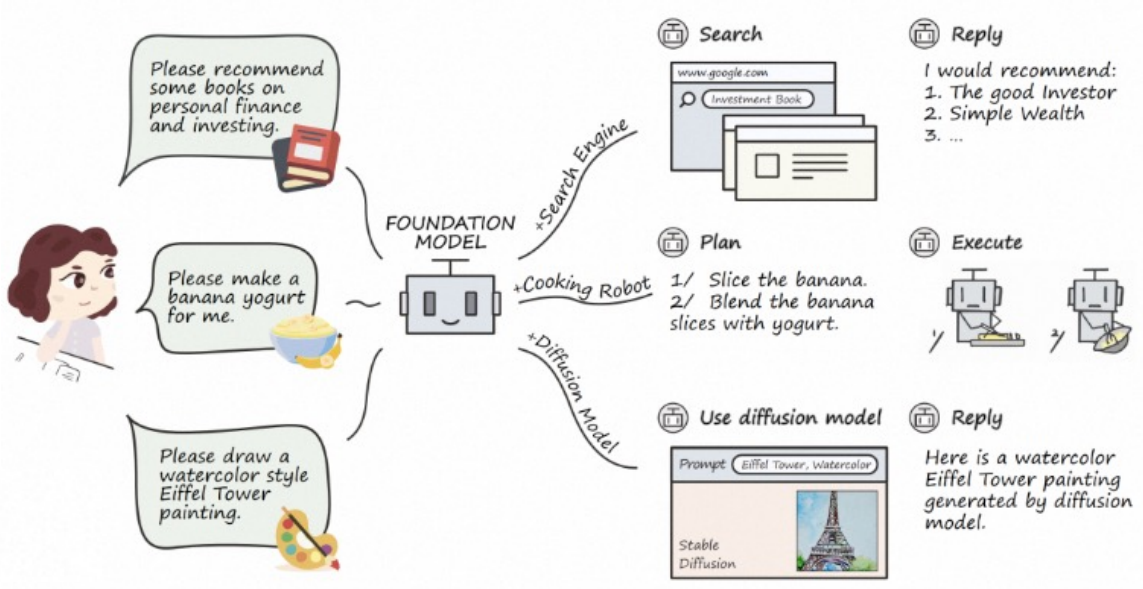
Advancing Tool-Augmented Large Language Models: *Integrating Insights from Errors in Inference Trees*

Sijia Chen*, Yibo Wang*, Yi-Feng Wu, Qing-Guo Chen, Zhao Xu,
Weihua Luo, Kaifu Zhang, Lijun Zhang

NeurIPS 2024

Background

- **Tool-augmented Large Language Models (LLMs)** leverage external tools, often in the form of APIs, to improve their reasoning capabilities on ***real-time knowledge and complex tasks.***



Motivation

- Advanced closed-source LLMs have demonstrated good tool usage capabilities

Ability	LLM	Call		Retrieve+Call		Plan+Retrieve+Call		Total	
		Correctness	Rouge	Correctness	Rouge	Correctness	Rouge	Correctness	Rouge
Zero-shot	Alpaca-7B	24.06%	0.0204	5.19%	0.0019	0.00%	0.086	15.19%	0.0318
	ChatGLM-6B	23.62%	0.2451	13.33%	0.2173	0.00%	0.1522	16.42%	0.2191
	GPT-3 Davinci	0.50%	0.1035	1.48%	0.091	0.00%	0.0156	0.57%	0.0814
	GPT-3.5-turbo	59.40%	0.4598	38.52%	0.3758	22.00%	0.3809	47.16%	0.4267
	GPT-4	63.66%	0.3691	37.04%	0.351	70.00%	0.4808	60.24%	0.3910
Fine-tuning	Lynx-7B	49.87%	0.4332	30.37%	0.2503	20.00%	0.3425	39.58%	0.3794

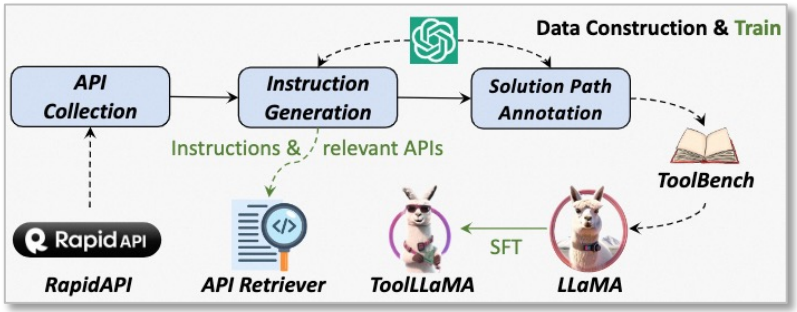
Table 3: Main results of different LLMs on the API-Bank evaluation system.

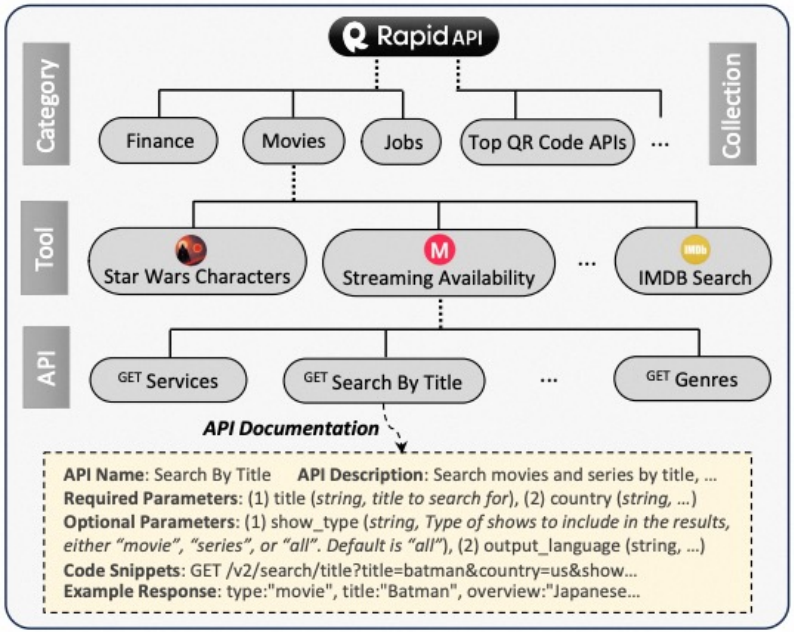
- Early tool learning research on open-source LLMs has certain limitations
  - ✓ Tools are ***limited or not real-world accessible***
  - ✓ Focus on scenarios where ***only one tool is used*** for one reasoning task
  - ✓ ***Simple reasoning and planning mechanism*** limits the tool-use potential of LLMs
  - ✓ Some studies ***do not use real responses*** from API execution for training

1. Li et al., API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs, 2023.

Motivation

- Key Reference Work —— ToolBench[1]



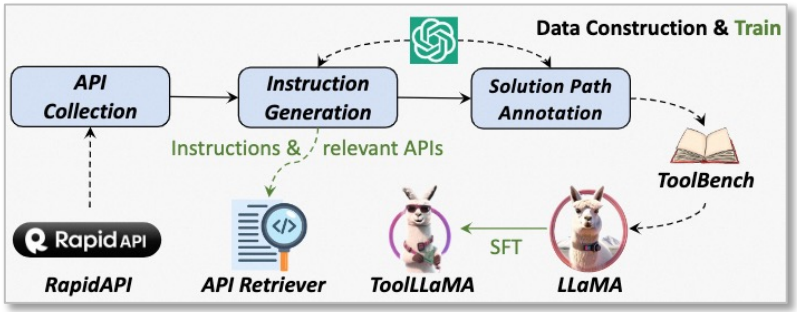


- Collect 3,451 real-world tools from RapidAPI Hub, which ***contain 16,464 APIs***
- Generate ***126,486 pairs of (instruction, expert annotated path) samples*** by using ChatGPT to create instructions that may call one or several APIs and annotate reasoning trajectories with real API calls
- Fine-tune LLaMA to create ToolLLaMA

1. Qin et al., Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023.

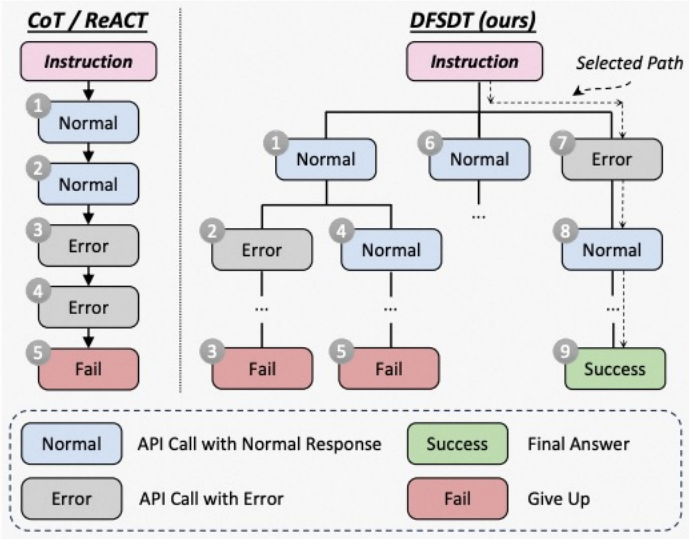
Motivation

➢ Key Reference Work —— ToolBench[1]



- Collect 3,451 real-world tools from RapidAPI Hub, which ***contain 16,464 APIs***
- Generat ***126,486 pairs of (instruction, expert annotated path) samples*** by using ChatGPT to create instructions that may call one or several APIs and annotate reasoning trajectories with real API calls
- Fine-tune LLaMA to create ToolLLaMA

Reasoning Mechanism：***Depth-First Search-based Decision Tree (DFSDT)***

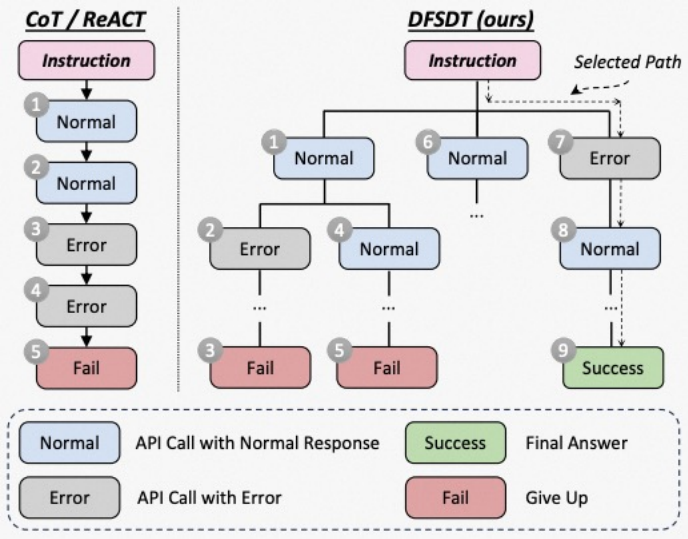


1. Qin et al., Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023.

Motivation

➢ Key Reference Work —— ToolBench[1]

Reasoning Mechanism：*Depth-First Search-based Decision Tree (DFSDT)*



- ToolLLaMA's performance still has room for improvement

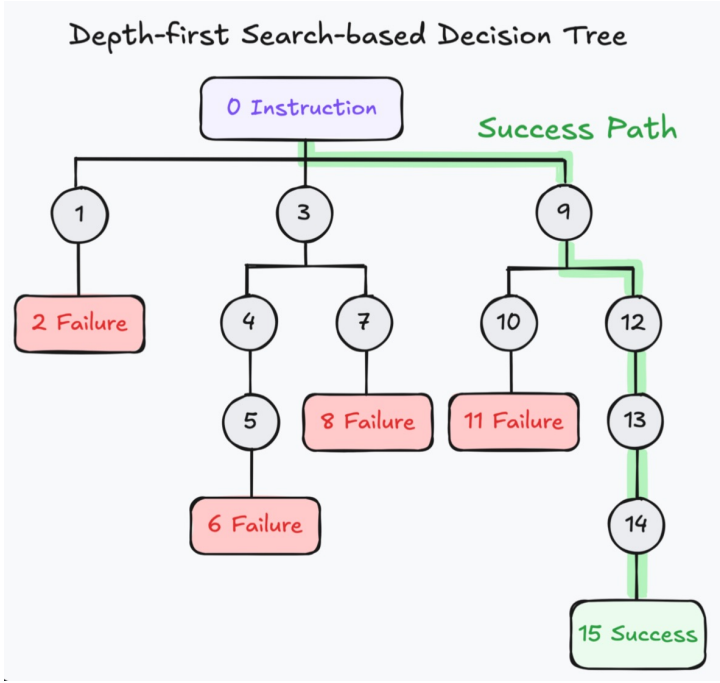
Model	Method	I1-Inst.		I1-Tool		I1-Cat.		I2-Inst.		I2-Cat.		I3-Inst.		Average	
		Pass	Win	Pass	Win	Pass	Win	Pass	Win	Pass	Win	Pass	Win	Pass	Win
ChatGPT	ReACT	41.5	-	44.0	-	44.5	-	42.5	-	46.5	-	22.0	-	40.2	-
	DFSDT	54.5	60.5	65.0	62.0	60.5	57.3	75.0	72.0	71.5	**64.8**	62.0	69.0	64.8	64.3
Claude-2	ReACT	5.5	31.0	3.5	27.8	5.5	33.8	6.0	35.0	6.0	31.5	14.0	47.5	6.8	34.4
	DFSDT	20.5	38.0	31.0	44.3	18.5	43.3	17.0	36.8	20.5	33.5	28.0	65.0	22.6	43.5
Text-Davinci-003	ReACT	12.0	28.5	20.0	35.3	20.0	31.0	8.5	29.8	14.5	29.8	24.0	45.0	16.5	33.2
	DFSDT	43.5	40.3	44.0	43.8	46.0	46.8	37.0	40.5	42.0	43.3	46.0	63.0	43.1	46.3
GPT4	ReACT	53.5	60.0	50.0	58.8	53.5	63.5	67.0	65.8	72.0	60.3	47.0	78.0	57.2	64.4
	DFSDT	60.0	**67.5**	**71.5**	**67.8**	**67.0**	**66.5**	79.5	**73.3**	**77.5**	63.3	**71.0**	**84.0**	**71.1**	**70.4**
Vicuna	ReACT & DFSDT	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Alpaca	ReACT & DFSDT	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ToolLLaMA	ReACT	25.0	45.0	29.0	42.0	33.0	47.5	30.5	50.8	31.5	41.8	25.0	55.0	29.0	47.0
	DFSDT	57.0	55.0	61.0	55.3	62.0	54.5	77.0	68.5	77.0	58.0	66.0	69.0	66.7	60.0
	DFSDT-Retriever	**64.0**	62.3	64.0	59.0	60.5	55.0	**81.5**	68.5	68.5	60.8	65.0	73.0	67.3	63.1

- ToolBench provides large-scale tree-like expert trajectories, ***but only the successful paths (positive samples) are used as training data during supervised fine-tuning***
  - Low data utilization
  - Insufficient exploration of the target space may lead to suboptimal strategies and limited generalization performance
  - Lack of fine-grained process supervision

1. Qin et al., Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023.

Our Method

➢ Dataset Construction——**ToolPreference** for ***multi-step reasoning with tools*** based on ToolBench



- *If we use a path-wise way to construct preference samples following previous researches:*

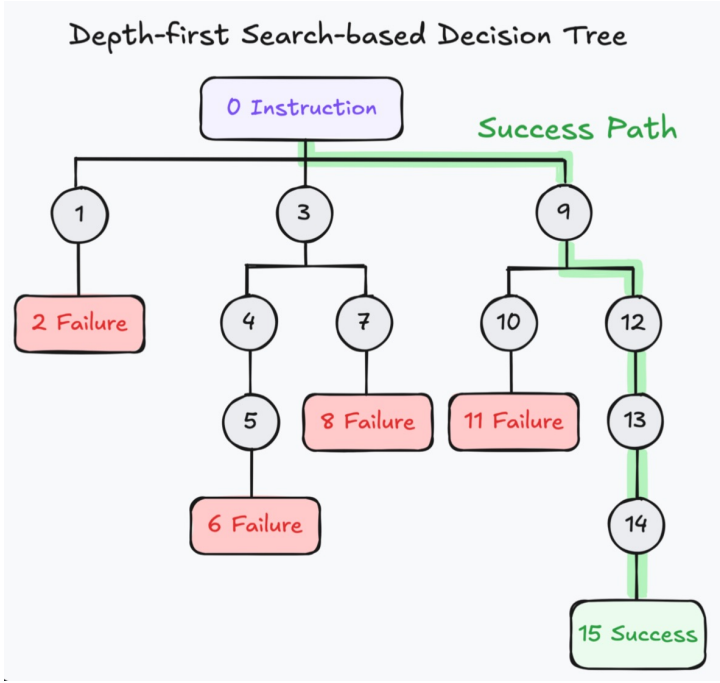
➢ **Theoretically,** this may cause the model to focus more on the final correct or incorrect responses to specific instructions, leading to poor generalization.

➢ **From an engineering perspective,** learning preference for the entire path conflicts with the model's **iterative reasoning mechanism**, making it unsuitable for implementing.

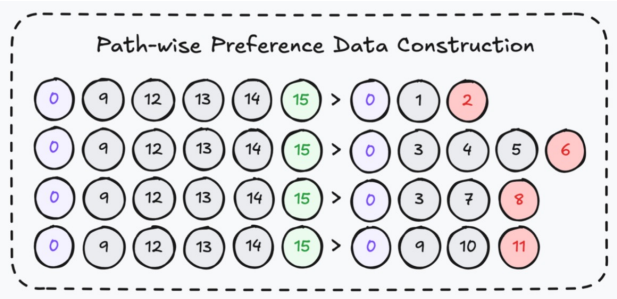
Our Method

Iterative Reasoning Mechanism: the model ***decides the next API call based on the response of last API execution***, rather than pre-planning all API calls at the start.

➢ Dataset Construction——**ToolPreference** for ***multi-step reasoning with tools*** based on ToolBench



- ***If we use a path-wise way to construct preference samples following previous researches:***

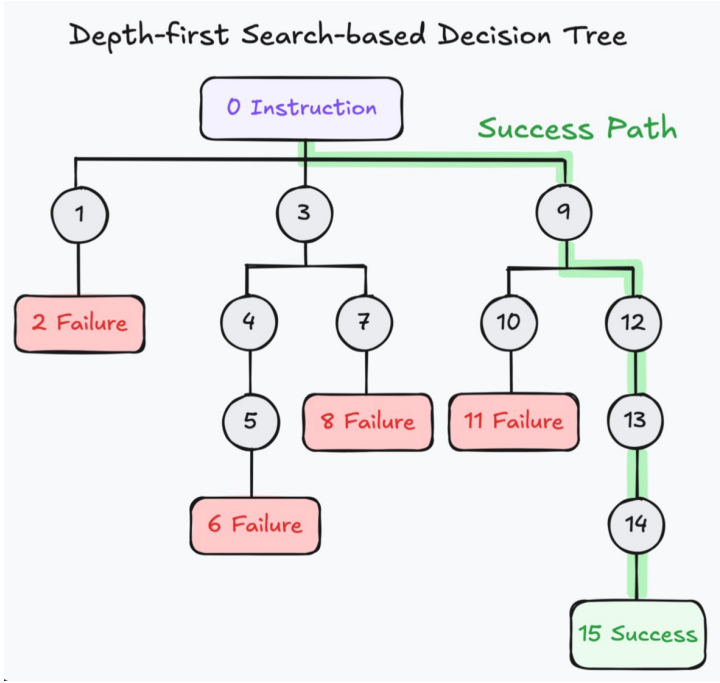


➢ **Theoretically,** this may cause the model to focus more on the final correct or incorrect responses to specific instructions, leading to poor generalization.

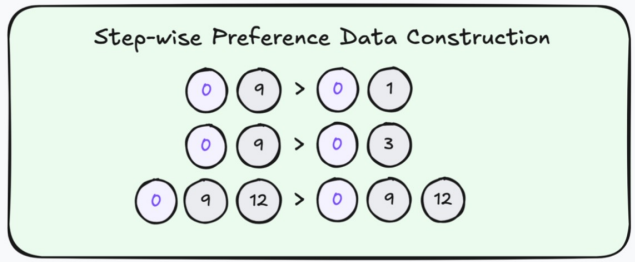
➢ **From an engineering perspective,** learning preference for the entire path conflicts with the model's **iterative reasoning mechanism**, making it unsuitable for implementing.

Our Method

➢ Dataset Construction——**ToolPreference** for ***multi-step reasoning with tools*** based on ToolBench



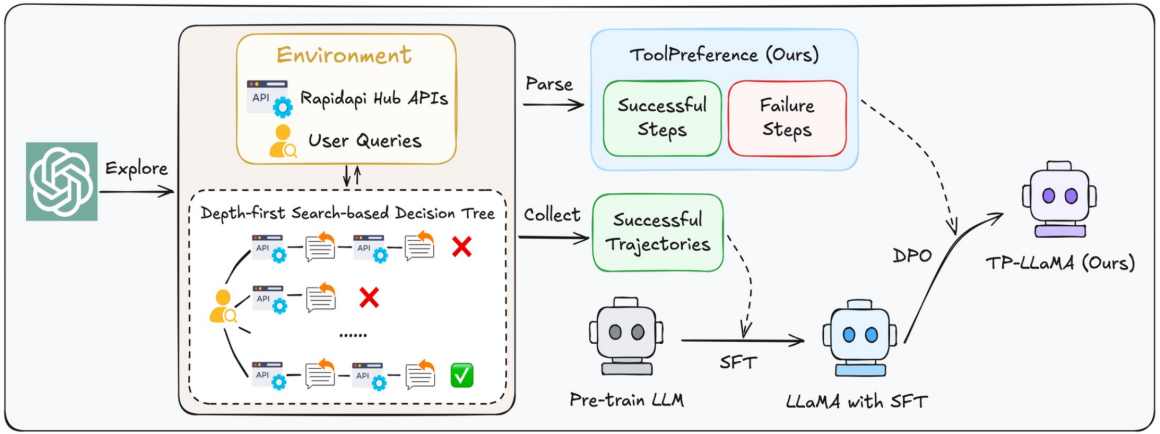
- Use a ***step-wise*** way to construct preference samples: ***backtrack along successful paths to parse branch nodes***



- Construct **69,393** pairs of preference samples, each formalized as **{Instruction, Input, Output}**:
  - **Instruction:** description of the DFSDT reasoning rules and available APIs documentation
  - **Input:** user query and historical inference information before the current step
  - **Output:** a pair of preferred and unpreferred steps

Our Method

➢ Aligning LLMs with experts' tool-usage preferences —— **Direct Preference Optimization (DPO)**



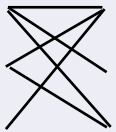
1. **Supervised Fine-Tuning (SFT)** is performed on the pre-trained LLM using only positive samples
2. ***Direct Preference Optimization (DPO)*** is performed with ToolPreference to obtain **our TP-LLaMA**

$$\mathcal{L}_{\mathrm{DPO}}(\pi_\theta;\pi_{\mathrm{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\mathrm{ref}}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\mathrm{ref}}(y_l|x)}\right)\right]$$

Our Method

- Evaluation Metrics
  - ✓ ***Pass Rate***: represents the proportion that the model successfully gives answers within a certain number of reasoning actions (set to 200 in our experiment)
  - ✓ ***Win Rate:*** measures the likelihood that the solution path provided by the test model is preferred over the reference solution path for the same instruction
- Test Setting
  - ✓ Evaluate on ***six test scenarios with different task complexity and generalization difficulty***

G1: single-tool
G2: intra-category multi-tools
G3: inter-category multi-tools



Instruction: unseen instructions for seen tools in training
Tool: unseen tools that belong to seen categories in training
Category: unseen tools that belong to unseen categories

Experiment Results

- Using **LLaMA-2-7B** as the Base Model for Training – **About Reasoning Performance**
  - ✓ **In terms of pass rate,** TP-LLaMA ***significantly outperforms the baselines,*** improving by more than ***12%*** on average compared to models without preference learning
  - ✓ **In terms of win rate,** TP-LLaMA also shows quite good performance, only slightly lower than ToolLLaMA in the G1-Category scenario
  - ✓ TP-LLaMA performs well in more challenging task scenarios. The pass rate increased by more than ***26%*** in the G3-Instruction scenario, proving that preference learning significantly ***enhances the model's ability to handle complex multi-tool tasks.***

Pass Rate

Model	G1-Ins.	G1-Tool	G1-Cat.	G2-Ins.	G2-Cat.	G3-Ins.	Avg
ChatGPT	0.52	0.55	0.60	0.51	0.51	0.21	0.48
Davinci	0.49	0.47	0.45	0.40	0.27	0.29	0.40
ToolLLaMA	0.54	0.60	0.62	0.47	0.54	0.17	0.49
LLaMA with SFT	0.47	0.53	0.72	0.48	0.63	0.35	0.53
TP-LLaMA (ours)	**0.55**	**0.65**	**0.80**	**0.62**	**0.67**	**0.61**	**0.65**

Win Rate

Model	G1-Ins.	G1-Tool	G1-Cat.	G2-Cat.	G2-Ins.	G3-Ins.	Avg
ChatGPT	-	-	-	-	-	-	-
Davinci	0.37	0.37	0.35	0.35	0.29	0.54	0.38
ToolLLaMA	0.55	0.53	**0.57**	0.56	0.52	0.68	0.57
LLaMA with SFT	0.54	0.51	0.56	0.65	0.57	0.81	0.61
TP-LLaMA (ours)	**0.56**	**0.59**	0.54	**0.70**	**0.64**	**0.86**	**0.65**

Experiment Results

- Using **LLaMA-2-7B** as the Base Model for Training – **About Reasoning Efficiency**
  - ✓ LLaMA with SFT requires an average of 32.06 reasoning steps, while TP-LLaMA needs only 22.62 steps, ***improving efficiency by 29.44%***
  - ✓ TP-LLaMA's reasoning efficiency is significantly better than the model trained only with successful trajectories -- through preference learning, the model is more likely to make the best decision first at each step of reasoning, thus avoiding exploring unnecessary suboptimal branches in the decision tree.

Table 2: Efficiency Results of TP-LLaMA. Imp denotes the improvement of TP-LLaMA over LLaMA with SFT in terms of the average steps.

Model	G1-Ins.	G1-Tool	G1-Cat.	G2-Ins.	G2-Cat.	G3-Ins.	Avg	Imp
LLaMA with SFT	32.82	34.60	31.45	31.98	35.05	26.44	32.06	-
TP-LLaMA (ours)	**24.54**	**24.19**	**23.85**	**23.98**	**23.53**	**15.61**	**22.62**	**29.44%**

Experiment Results

- Ablation Experiment: Using Mistral-7B、Qwen1.5-7B、Gemma-7B as Base Models

Table 3: Ablation Performance Experiment Results. Avg represents the average pass rate or win rate of the 6 test scenarios. A win rate higher than 50% means the model performs better than ChatGPT+DFSDT.

Pass Rate							
Model	**G1-Ins.**	**G1-Tool**	**G1-Cat.**	**G2-Ins.**	**G2-Cat.**	**G3-Ins.**	**Avg**
Mistral with SFT	0.70	0.43	0.42	0.53	0.46	0.27	0.47
TP-LLaMA (Mistral)	**0.71**	**0.53**	**0.55**	**0.70**	**0.64**	**0.57**	**0.62**
Qwen with SFT	0.69	0.51	0.51	0.66	0.55	0.49	0.57
TP-LLaMA (Qwen)	**0.77**	**0.53**	**0.60**	**0.72**	**0.61**	**0.65**	**0.65**
Gemma with SFT	0.67	0.44	0.49	0.47	0.44	0.29	0.47
TP-LLaMA (Gemma)	**0.80**	**0.48**	**0.61**	**0.70**	**0.65**	**0.68**	**0.65**
Win Rate							
Model	**G1-Ins.**	**G1-Tool**	**G1-Cat.**	**G2-Cat.**	**G2-Ins.**	**G3-Ins.**	**Avg**
Mistral with SFT	0.52	0.47	0.55	0.61	0.61	0.72	0.58
TP-LLaMA (Mistral)	**0.53**	**0.50**	**0.57**	**0.62**	**0.64**	**0.74**	**0.60**
Qwen with SFT	0.53	0.52	0.52	0.64	0.66	0.75	0.60
TP-LLaMA (Qwen)	**0.54**	**0.54**	**0.58**	**0.66**	**0.67**	**0.81**	**0.63**
Gemma with SFT	0.58	0.54	0.53	0.50	0.62	0.73	0.58
TP-LLaMA (Gemma)	**0.61**	**0.57**	**0.58**	**0.65**	**0.67**	**0.75**	**0.64**

Table 4: Ablation Efficiency Experiment Results. Imp denotes the improvement of TP-LLaMA over LLaMA with SFT in terms of the average steps.

Model	**Average Number of Steps in One Successful Path**							
	G1-Ins.	**G1-Tool**	**G1-Cat.**	**G2-Ins.**	**G2-Cat.**	**G3-Ins.**	**Avg**	**Imp**
Mistral with SFT	28.92	26.65	30.22	25.69	26.58	25.24	27.22	-
TP-LLaMA (Mistral)	**25.30**	**25.01**	**23.36**	**23.51**	**20.74**	**16.42**	**22.39**	**17.74%**
Qwen with SFT	35.74	34.66	36.85	32.74	36.18	37.93	35.68	-
TP-LLaMA (Qwen)	**25.12**	**23.83**	**24.49**	**23.84**	**26.92**	**22.18**	**24.40**	**31.61%**
Gemma with SFT	27.49	22.77	24.10	18.70	20.52	21.19	22.46	-
TP-LLaMA (Gemma)	**17.15**	**13.88**	**15.91**	**13.63**	**13.30**	**15.94**	**14.97**	**33.35%**

Preference learning improves the performance of all the models. The effectiveness of our framework is independent of the model itself!

Advancing Tool-Augmented Large Language Models: Integrating Insights from Errors in Inference Trees

Sijia Chen[1,2,*] **Yibo Wang**[1,2,*] **Yi-Feng Wu**[3] **Qing-Guo Chen**[3]
Zhao Xu[3] **Weihua Luo**[3] **Kaifu Zhang**[3] **Lijun Zhang**[1,4,2,†]

[1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[2]School of Artificial Intelligence, Nanjing University, Nanjing, China
[3]Alibaba International Digital Commerce [4]Pazhou Laboratory (Huangpu), Guangzhou, China
{chensj, wangyb, zhanglj}@lamda.nju.edu.cn
{yixin.wyf, qingguo.cqg, changgong.xz,
weihua.luowh, kaifu.zkf}@alibaba-inc.com

Abstract

Tool-augmented large language models (LLMs) leverage tools, often in the form of APIs, to improve their reasoning capabilities on complex tasks. This enables them to act as intelligent agents interacting with the real world. The recently introduced ToolLLaMA model by Qin et al. [2023] utilizes the depth-first search-based decision tree (DFSDT) mechanism for multi-step reasoning with 16000+ real-world APIs, effectively enhancing the performance of tool-augmented LLMs compared to traditional chain reasoning mechanisms. However, their approach only employs successful paths from decision trees (also called inference trees) for supervised fine-tuning (SFT), missing out on the potential learning opportunities from failed paths. Inspired by this, we propose an inference trajectory optimization framework based on preference learning to address this limitation. We first introduce a novel method for constructing preference data from tree-like expert trajectories, which leverages the previously ignored failed explorations in the decision trees. Specifically, we generate a step-wise preference dataset, ToolPreference, from the ToolBench dataset for tool learning. In the subsequent training phase, we first fine-tune the LLM with successful tool-usage expert trajectories and then apply direct preference optimization (DPO) with ToolPreference to update the LLM's policy, resulting in our ToolPrefer-LLaMA (TP-LLaMA) model. This approach not only enhances the utilization of original expert data but also broadens the learning space of the model. Our experiments demonstrate that by obtaining insights from errors in inference trees, TP-LLaMA significantly outperforms the baselines across almost all test scenarios by a large margin and exhibits better generalization capabilities with unseen APIs. At the same time, TP-LLaMA has also demonstrated superior reasoning efficiency compared to the baselines, making it more suitable for complex tool-usage reasoning tasks.

1 Introduction

In recent years, large language models (LLMs) have exhibited impressive capabilities in various areas, including language understanding and generation, multi-modal content learning and reasoning, and even embodied intelligence task processing [Brown et al., 2020, Zeng et al., 2023, Alayrac et al., 2022, Li et al., 2023, Lu et al., 2024, Cao et al., 2024a,b, Mazzaglia et al., 2024]. Despite these notable strengths, these models still face significant challenges, such as a lack of access to real-time information [Komeili et al., 2021] and difficulties in precise mathematical tasks [Patel et al., 2021, Lu et al., 2023b]. The development of tool-augmented LLMs tackles these challenges by enabling LLMs to interact with external tools (often in the form of APIs), significantly enhancing their capabilities. This advancement allows LLMs to serve as efficient intermediaries between users and a large ecosystem of applications. Notably, tool-augmented LLMs based on the ChatGPT [Brown et al., 2020] and GPT-4 [Achiam et al., 2023] have achieved outstanding results by using few-shot or zero-shot prompts to activate the LLM's inherent tool-usage abilities [Deng et al., 2023, Lin et al., 2024, Lu et al., 2023a]. Despite this progress, some studies demonstrate that open-source LLMs still exhibit a significant gap in their capacity to utilize external tools compared to state-of-the-art (SOTA) closed-source models like GPT-4 [Liu et al., 2024, Wang et al., 2024b]. To bridge this gap, aligning these open-source LLMs with tool-usage downstream tasks is essential.

Currently, most efforts to align open-source LLMs with tool-usage downstream tasks rely on supervised fine-tuning (SFT) with expert trajectory datasets, which trains LLMs to learn strategies for subsequent actions based on previous actions and observations [Patil et al., 2023, Schick et al., 2023]. Early studies in this field typically have limitations such as a restricted variety of APIs, the reliance on single-tool scenarios, and the use of simple reasoning methods [Wei et al., 2022, Yao et al., 2023, Patil et al., 2023]. The recent work by Qin et al. [2023], which focuses on the scene of LLM's multi-step reasoning with external tools, solves the above limitations. They introduce an instruction tuning dataset called ToolBench, which includes over 16,000 real-world APIs and various realistic instructions, along with expert trajectories annotated by ChatGPT based on a depth-first search-based decision tree (DFSDT) reasoning mechanism. They then perform SFT training on LLaMA with this dataset to create the ToolLLaMA model, which shows remarkable performance. However, ToolLLaMA's training is still based on expert behavior cloning, potentially limiting exploration of the target space and leading to suboptimal strategies. Additionally, although their expert trajectories are structured as DFS trees, only successful trajectories are utilized in the SFT training, which neglects valuable insights from failed attempts and results in low data utilization.

As the saying goes, "a fall into a pit, a gain in your wit", effective human learning involves not only drawing lessons from success but also from failures. Inspired by this, we propose a new inference trajectory optimization framework for developing tool-augmented LLMs as illustrated in Figure 1, which enhances the tool learning process by incorporating previously ignored failure exploration information via preference learning. Specifically, using the tree-like expert trajectories from ToolBench [Qin et al., 2023], we first parse each pair of branch nodes along the successful trajectory in the decision tree into a preference sample pair, thereby constructing a novel step-wise tool-usage preference dataset named *ToolPreference*. Subsequently, after conducting SFT training on the pre-trained LLM with successful trajectories, we employ the direct preference optimization (DPO) method [Rafailov et al., 2023] with the ToolPreference dataset to further align the LLM with tool-usage downstream tasks, and thus obtain our model, named *ToolPrefer-LLaMA (TP-LLaMA)*. Our strategy improves the utilization of expert data and simultaneously broadens the learning space.

Our experiments are conducted on the test tasks from ToolBench. To evaluate the performance, we adopt two metrics: the *pass rate*, which measures the probability of the model successfully providing an answer within limited steps; and the *win rate*, which quantifies the likelihood that the evaluator will prefer the model's responses. From the experiment results, we have the following findings:

- Across all test scenarios, TP-LLaMA consistently surpasses ToolLLaMA and other baselines, with an average pass rate improvement of at least 12% and a win rate that outperforms nearly all other models by an average of 4%. These results demonstrate that learning from failed attempts can significantly enhance the decision-making ability of LLMs. Additionally, our model shows superior generalization to unseen APIs.
- Efficiency experiments show that our model requires an average of only 22.62 steps for DFSDT inference, compared to 32.06 steps for the SFT model. This enhancement stems from our method's ability to avoid unnecessary branch explorations in DFSDT reasoning.
- Our ablation experiments verify that the effectiveness of our preference dataset and inference trajectory optimization framework has nothing to do with the base model itself. Better results can still be obtained after replacing the base model with Mistral-7B [Jiang et al., 2023], Qwen1.5-7B [Bai et al., 2023], and Gemma-7B [Team et al., 2024].

In summary, this work aims to enhance the performance of LLMs on multi-step reasoning with external tools by integrating insights from errors in tree-like reasoning trajectories and employing step-wise preference pairs for preference learning. Our key contributions include: (i) The careful design and construction of a new tool-usage preference dataset — ToolPreference, providing a valuable resource for the community; (ii) The proposal of using ToolPreference to optimize the LLM's tool-usage ability with the DPO training, along with the development of the TP-LLaMA model; (iii) Extensive experimental evaluations and in-depth analyses of the TP-LLaMA model, providing evidence of its effectiveness and validating its performance across various dimensions.

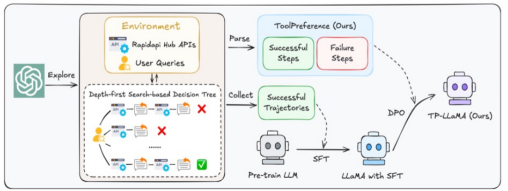


Figure 1: Our Inference Trajectory Optimization Framework.

Thanks!