ToolkenGPT: Augmenting Frozen Language Models with Massive Tools via Tool Embeddings

Shibo Hao, Tianyang Liu, Zhen Wang, Zhiting Hu
{s5hao, til040, zhw085, zhh019}@ucsd.edu

Background

Large language models (LLMs) have shown remarkable intelligence, but they suffer from important limitations hindering a broader deployment, e.g., making factual or arithmetical errors, inability to interact with world…

Augmenting LLMs with external tools (e.g. calculator, database…) is a promising solution.

Example tools in ChatGPT plugin: Wolfram (calculation), Instacart (grocery delivery), Kayak (flight search), …

Most previous research or products (including ChatGPT plugin) relies on in-context learning (ICL), i.e. prompting the LLM with descriptions and demonstrations of tools.

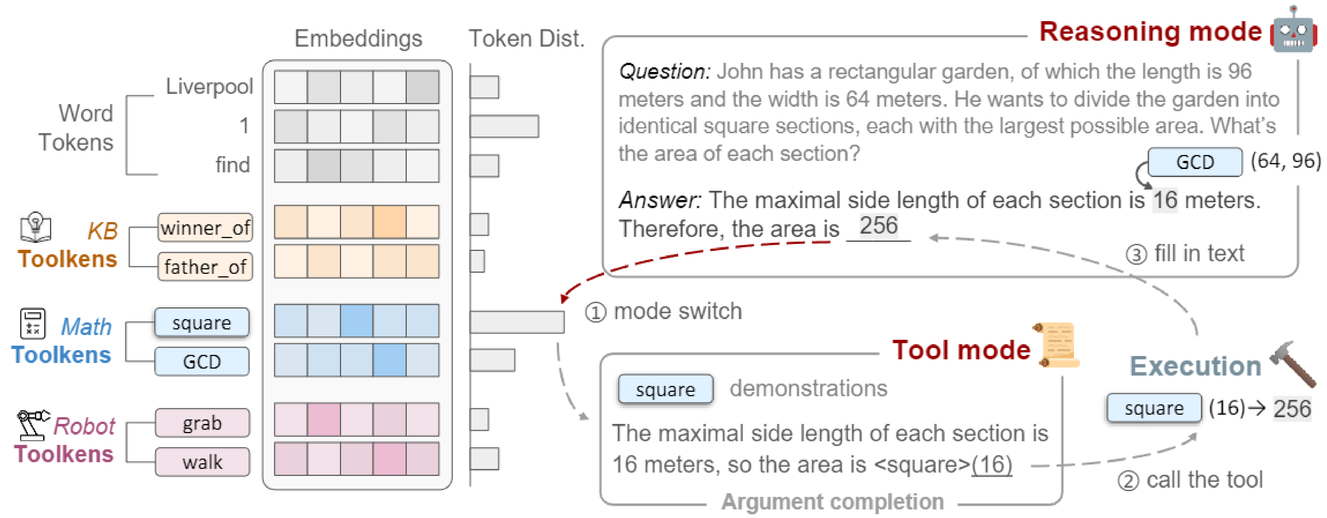
However, due to the limited context length, ICL can not learn from extensive data and handle massive tools. We propose an alternative framework, **ToolkenGPT**, to overcome these challenges.

Tool Learning Paradigms	Frozen LMs	Massive Tools	Plug-&-Play	Ability to Use Extensive Data
Fine-tuning [e.g., 53, 47]	✗	✗	✗	✓
In-context learning [e.g., 65, 50, 7]	✓	✗	✓	✗
ToolkenGPT (**Ours**)	✓	✓	✓	✓

Framework

Our approach **represents each tool as a token ("toolken")** and learns an embedding for it.

- In the "reasoning mode", the LLM generates text as usual, except that plugged-in toolkens are also considered for the next token.
- Once a toolken is predicted, the LLM switch to the "tool mode", which provides a few demonstrations of the selected tool to complete the arguments.
- Finally, the call is executed by an external tool, and the results are sent back to the text to in the reasoning mode.



Training. We append toolken embeddings to the language model head and train them. Given a word token sequence s and its corresponding mixed sequence s' of word tokens and toolkens:

$$\mathcal{L} = \sum_{(s,s')\in\mathcal{D}} \sum_{i=1}^{N} -\log P(t'_i|t_{<i}) \mathbb{1}_{t'_i \neq \texttt{[N/A]}}$$

Example: s = […, "area", "is", "2", "5", "6", "square", …]
s'= […, "area", "is", " [square] ", "[N/A]", "[N/A]", "square", …]

Experiments

1. Numerical Reasoning. We consider 4 operators (+, −, ×, ÷) on GSM8K-XL and 13 more complex operators for FuncQA.

"… so the area is 256 square feet" square(16)=256

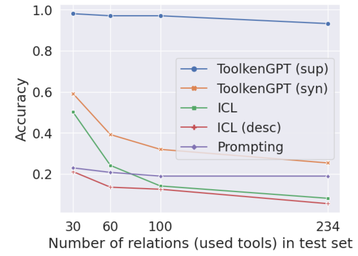
- Outperforms ReAct (an ICL-based method) and even surpasses ChatGPT with Llama-33B (an opensource LLM)

Method	GSM8K-XL (4)	FuncQA (13)	
		One-Hop	Multi-Hops
0-shot ChatGPT	0.17	0.55	0.09
CoT [62]	0.18	0.20	0.03
ReAct [65]	0.32	0.57	0.06
ToolkenGPT (Ours)	**0.33**	**0.73**	**0.15**

2. Knowledge-based QA. We consider each relation in Wikidata as a tool to enable knowledge base query.

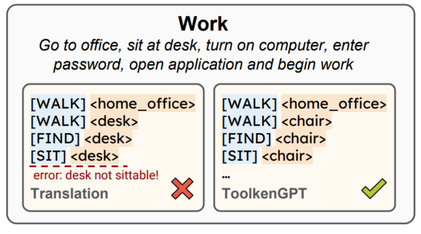
"…Liverpool was the champion" winner("FA Cup")="Liverpool"

- Outperforms ICL with even only synthetic training data.
- Remains satisfactory performance when the questions require massive tools to answer.



3. Embodied Plan Generation. Given household tasks, We generate plans for embodied agents with ToolkenGPT. Admissible actions and objects are taken as toolkens.

Method	Grounding	Executable	Success	Success (R)
In-context Learning	0.74	0.42	0.20	0.30
+ Translation [22]	**1.00**	0.52	0.24	0.32
+ Grounded Decoding [24]	**1.00**	0.66	0.38	0.42
ToolkenGPT (Ours)	**1.00**	**0.82**	**0.68**	**0.70**



- Improves executable rate and successful rate
- learning deeper semantics instead of surface text.

arXiv GitHub