Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning

在RLVR背景下，如何做SEARCH-AND-REASONING?

Bowen Jin[1], **Hansi Zeng**[2], **Zhenrui Yue**[1], **Jinsung Yoon**[3], **Sercan Ö. Arık**[3], **Dong Wang**[1]
Hamed Zamani[2], **Jiawei Han**[1]
[1] Department of Computer Science, University of Illinois at Urbana-Champaign
[2] Center for Intelligent Information Retrieval, University of Massachusetts Amherst
[3] Google Cloud AI Research
{bowenj4,zhenrui3,dwang24,hanj}@illinois.edu, {hzeng, zamani}@cs.umass.edu

简介

开源代码：HTTPS://GITHUB.COM/PETERGRIFFINJIN/SEARCH-R1

本文提出了Search-R1，用RLVR的方法，同时提升LLM的推理和使用搜索引擎的能力。我们需要关注作者如何设计reward function，以及额外注意的一点是由于rollout中含有搜索引擎返回的检索内容，在计算loss (包括kl项)时要mask这些token，它们并不参与LLM的参数优化。

背景

LLM与搜索引擎（search engine）结合可以扩展其内部知识，如何结合呢？一种方法是RAG，通过搜索引擎的检索结果来扩展prompt；另一种是把搜索引擎看作一种tool，让LLM学会使用search tool。
让LLM使用tool，最简单的方法是写prompt template，比如解释下search tool可以做什么，再举几个使用tool的prompt的例子，类似CoT。还可以对LLM做fine-tuning，训练它学会使用search tool，本文聚焦用RL做tuning，既让LLM提升推理能力又学会使用search tool

实验设置

训练集：NQ 和 HOTPOTQA

- 框架：verl，实验对象：Qwen2.5-3B/7B (Base/Instruct)，强化学习算法：PPO和GRPO
- 通过prompt约束LLM的response用标签分隔，<search>和</search>触发search，检索内容用<information>和</information>标记，LLM的推理内容则用<think>和</think>标记，最终答案用<answer>和</answer>标记，并且可以多次调用search tool
- ORM形式的RLVR reward function，看answer是否正确，不包含format reward：

$$r_\phi(x, y) = \mathrm{EM}(a_{\text{pred}}, a_{\text{gold}}).$$

训练流程

不要看图中的REWARD MODEL，其实是RULE BASED REWARD FUNCTION，典型的RLVR

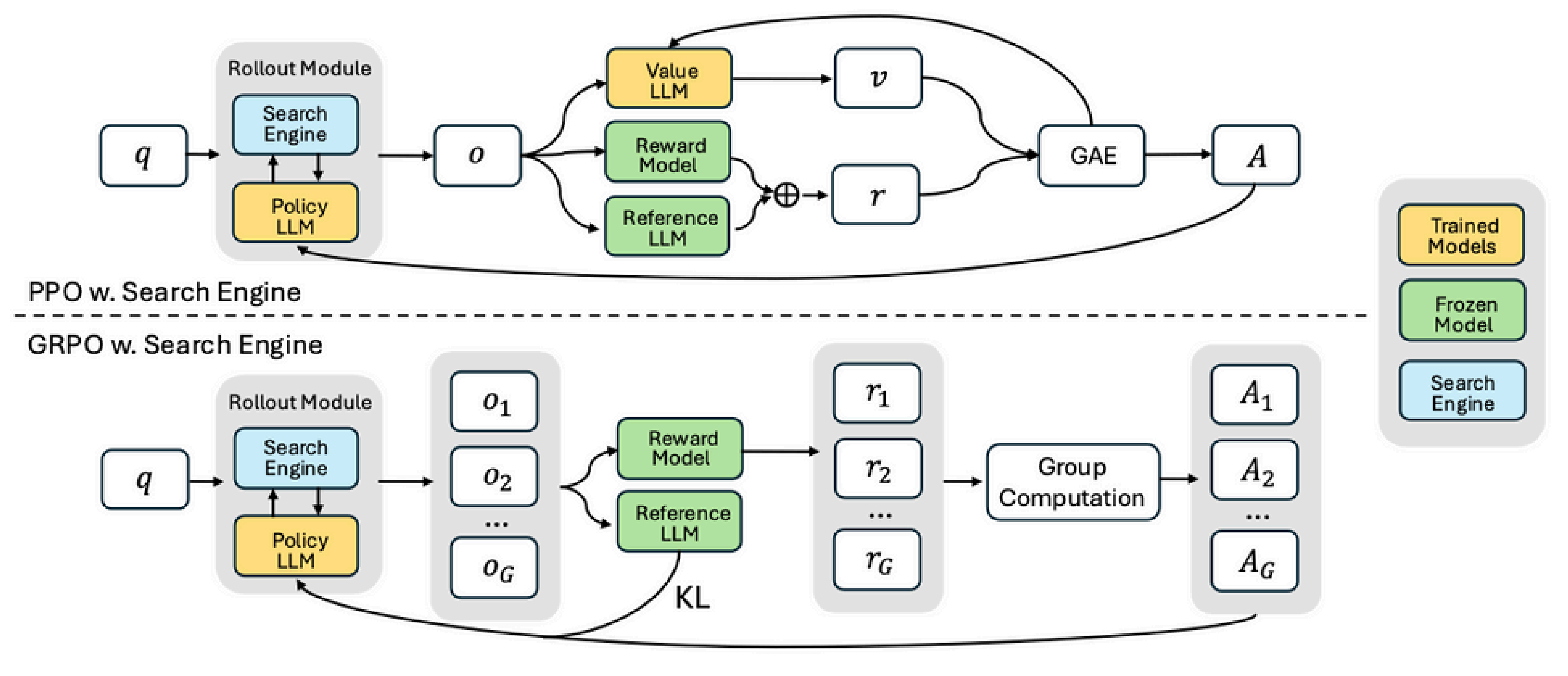

Figure 1: Demonstration of PPO and GRPO training with the search engine (SEARCH-R1). During the rollout, LLMs can conduct multi-turn interactions with the search engine.

Algorithm 1 LLM Response Rollout with Multi-Turn Search Engine Calls

Require: Input query x, policy model π_θ, search engine $\mathcal{R}$, maximum action budget B.
Ensure: Final response y.

Initialize rollout sequence $y \leftarrow \emptyset$
Initialize action count $b \leftarrow 0$
while $b < B$ **do**
  Initialize current action LLM rollout sequence $y_b \leftarrow \emptyset$
  while True **do**
    Generate response token $y_t \sim \pi_\theta(\cdot \mid x, y + y_b)$
    Append y_t to rollout sequence $y_b \leftarrow y_b + y_t$
    if y_t in [</search>, </answer>, <eos>] **then** break
    end if
  end while
  $y \leftarrow y + y_b$
  if <search> </search> detected in y_b **then**
    Extract search query $q \leftarrow \text{Parse}(y_b,$ <search>, </search>$)$
    Retrieve search results $d = \mathcal{R}(q)$
    Insert d into rollout $y \leftarrow y +$ <information>d</information>
  else if <answer> </answer> detected in y_b **then**
    return final generated response y
  else
    Ask for rethink $y \leftarrow y +$ "My action is not correct. Let me rethink."
  end if
  Increment action count $b \leftarrow b + 1$
end while
return final generated response y

思考

本文是3月份的工作，算是比较早用RLVR做提升LLM的推理和TOOL UUSING的工作，并且没有选择大众化的数学/编程领域，而是聚焦搜索引擎工具，重点是理解ROLLOUT过程，如果生成了<SEARCH>...</SEARCH>，则生成中断，系统调用搜索引擎返回检索结果，用<INFORMATION>...</INFORMATION>包裹，追加到RESPONSE，然后LLM继续生成，直到序列长度达到最大阈值或者生成了<ANSWER>...</ANSWER>才终止。

部分实验结果

Methods	General QA			Multi-Hop QA				
	NQ†	TriviaQA*	PopQA*	HotpotQA†	2wiki*	Musique*	Bamboogle*	Avg.
Qwen2.5-7b-Base/Instruct								
Direct Inference	0.134	0.408	0.140	0.183	0.250	0.031	0.120	0.181
CoT	0.048	0.185	0.054	0.092	0.111	0.022	0.232	0.106
IRCoT	0.224	0.478	0.301	0.133	0.149	0.072	0.224	0.239
Search-o1	0.151	0.443	0.131	0.187	0.176	0.058	0.296	0.206
RAG	0.349	0.585	0.392	0.299	0.235	0.058	0.208	0.304
SFT	0.318	0.354	0.121	0.217	0.259	0.066	0.112	0.207
R1-base	0.297	0.539	0.202	0.242	0.273	0.083	0.296	0.276
R1-instruct	0.270	0.537	0.199	0.237	0.292	0.072	0.293	0.271
Search-R1-base	**0.480**	**0.638**	**0.457**	**0.433**	0.382	**0.196**	**0.432**	**0.431**
Search-R1-instruct	0.393	0.610	0.397	0.370	**0.414**	0.146	0.368	0.385

@机器爱学习