

# REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

ArXiv – Project – STAN49

# REACT – (Reasoning + Acting)

- On a high level, the ReAct framework enables LLM's to utilize *tools* or *functions* while also extends the concept of chain-of-thought to chain-of-thought-action-observing.
- OpenAI's plugin feature builds on the ReAct framework.

# REACT – (Reasoning + Acting)

- As the name suggests, the ReAct framework enables the LLM for reasoning and action taking.
- Best demonstrated via example.

## (1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

- The LLM has then 2 tools at its disposal:
  - A Wikipedia search tool, Search[entity]
  - A Wikipedia lookup tool (ctrl+F sim), lookup[string]
- These tools and Finish[] makes up the actions the Agents can take.

## (1d) ReAct (Reason + Act)

**Thought 1:** I need to search Apple Remote and find the program it was originally designed to interact with.

**Act 1:** Search[Apple Remote]

**Obs 1:** The Apple Remote is a remote control introduced in October 2005 by Apple ... originally designed to control the Front Row media center program ...

**Thought 2:** Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

**Act 2:** Search[Front Row]

**Obs 2:** Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports', 'Front Row (software)', ...]

**Thought 3:** Front Row is not found. I need to search Front Row (software) .

**Act 3:** Search[Front Row (software)]

**Obs 3:** Front Row is a discontinued media center software ...

**Thought 4:** Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

**Act 4:** Finish[keyboard function keys]



<https://arxiv.org/pdf/2210.03629.pdf>

# REACT – How?

- How does the next-word prediction LLM model know when to think and to take action?
- Prompting...
- In the paper, the "agent" is fed with a few-shot prompting. i.e it is fed with a few examples of the thought-trajectory:

```
Question      What is the elevation range for the area that the eastern sector of the
               Colorado orogeny extends into?
Thought 1     I need to search Colorado orogeny, find the area that the eastern sector
               of the Colorado orogeny extends into, then find the elevation range of the
               area.
Action 1      Search[Colorado orogeny]
Observation 1 The Colorado orogeny was an episode of mountain building (an orogeny) in
               Colorado and surrounding areas.
Thought 2     It does not mention the eastern sector. So I need to look up eastern
               sector.
Action 2      Lookup[eastern sector]
Observation 2 (Result 1 / 1) The eastern sector extends into the High Plains and is called
               the Central Plains orogeny.
Thought 3     The eastern sector of Colorado orogeny extends into the High Plains. So I
               need to search High Plains and find its elevation range.
Action 3      Search[High Plains]
Observation 3 High Plains refers to one of two distinct land regions
Thought 4     I need to instead search High Plains (United States).
Action 4      Search[High Plains (United States)]
```

# Implementation – LangChain – ArXiv BOT - Tools

- Building LLM-agents in the python package LangChain is fairly simple
- The ArXiv Chatbot has access to two tools.

## TOOL 1: Search and Retrieve papers

***name*** = "paper retrieval"

***description*** = "Useful for when you need to get papers from arxiv. Input should be a search word. Always include the pdf url link of your answer and a short summary of the paper summary."

## TOOL 2: Summarizer

***name*** = "summarizer"

***description*** = "Useful for when you need to summarize papers. It should be a paper title."

- The name and description are inserted in the prompt. It is based on the description that the LLM reason about whether or not to use that tool.

# ArXiv BOT – Tools – Under the Hood

## **TOOL 1: *Under the hood* "paper retrieval"**

1. Takes a search word as input.
2. Using the arXiv- API it retrieves the k most relevant papers.
- 3a. Outputs the title and summary of the summary along with pdf-link for each paper.
- 3b. This tool also stores the pdf-link, the title, an embedded version of the titles and the summary for each paper that was retrieved in a csv-database.

## **TOOL 2: *Under the hood*: "summarizer"**

1. Takes a title as input.
2. Converts this title to an embedding
3. Performs similarity search between this embedding and the embeddings in the csv-database.
4. Retrieves pdf to the most similar title.
5. Chunk the pdf by page
6. Conduct "map-reduce" summarization.
  - a. per page bullet point summary
  - b. summary of all bullet-points

# ArXiv BOT – Prompt

- SystemMessage Prompt:

Assistant is a large language model trained by OpenAI.

Assistant is designed to be able to assist with a wide range of tasks, from answering simple questions to providing in-depth explanations and discussions on a wide range of topics. As a language model, Assistant is able to generate human-like text based on the input it receives, allowing it to engage in natural-sounding conversations and provide responses that are coherent and relevant to the topic at hand.

Assistant is constantly learning and improving, and its capabilities are constantly evolving. It is able to process and understand large amounts of text, and can use this knowledge to provide accurate and informative responses to a wide range of questions. Additionally, Assistant is able to generate its own text based on the input it receives, allowing it to engage in discussions and provide explanations and descriptions on a wide range of topics.

Overall, Assistant is a powerful system that can help with a wide range of tasks and provide valuable insights and information on a wide range of topics. Whether you need help with a specific question or just want to have a conversation about a particular topic, Assistant is here to assist.

# ArXiv BoT – Prompt

- HumanMessage Prompt:

```
TOOLS
-----
Assistant can ask the user to use tools to look up information that may be helpful in answering the users original question.
The tools the human can use are:

> paper retrieval: Useful for when you need to get papers from arxiv. Input should be a search word.
                    Always include the pdf url link of your answer and a short summary of the paper summary.

> summarizer: Useful for when you need to summarize papers. Input should be a paper title.

RESPONSE FORMAT INSTRUCTIONS
-----

When responding to me, please output a response in one of two formats:

**Option 1:**
Use this if you want the human to use a tool.
Markdown code snippet formatted in the following schema:

```json
{{
  "action": string, \ The action to take. Must be one of paper retrieval, summarizer
  "action_input": string \ The input to the action
}}
```

**Option #2:**
Use this if you want to respond directly to the human. Markdown code snippet formatted in the following schema:

```json
{{
  "action": "Final Answer",
  "action_input": string \ You should put what you want to return to use here
}}
```

USER'S INPUT
-----

Here is the user's input (remember to respond with a markdown code snippet of a json blob with a single action, and NOTHING
else):

{input}
```



# ArXiv BoT – Code

- Check Code!