ToolGPT

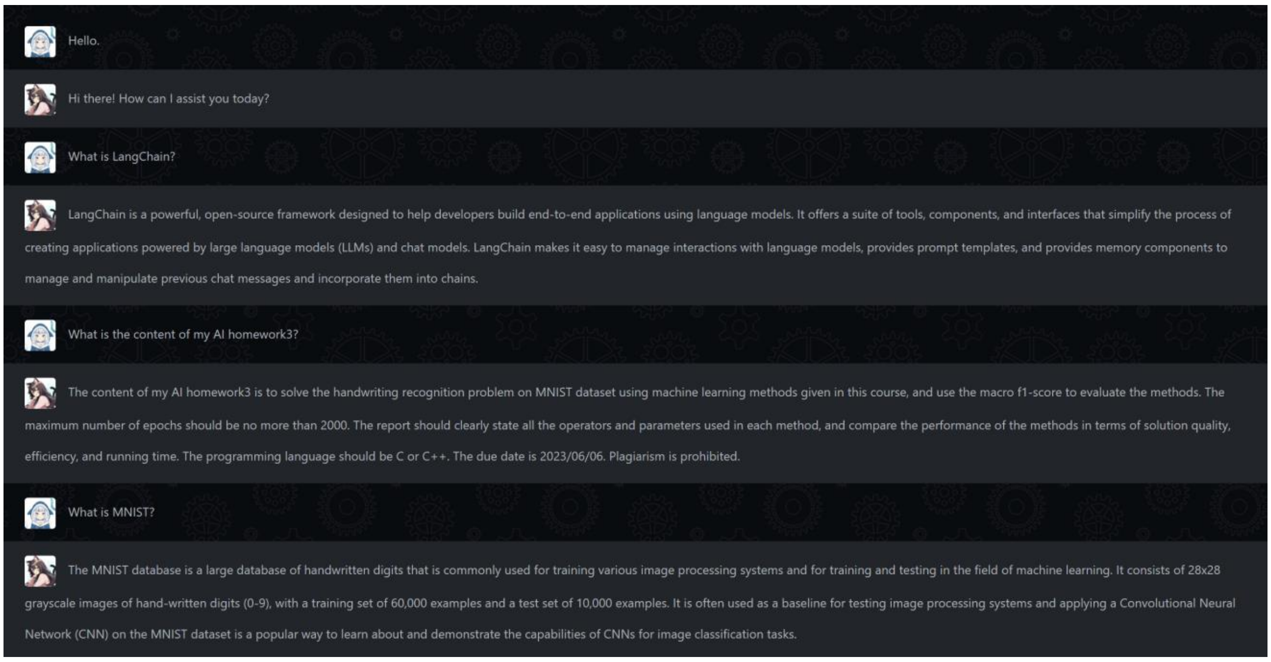
June 10, 2023

羅心義
資工二甲

尤奕翔
資工二甲

黃力宏
資工二甲

羅奕晟
資工二甲



Hello.

Hi there! How can I assist you today?

What is LangChain?

LangChain is a powerful, open-source framework designed to help developers build end-to-end applications using language models. It offers a suite of tools, components, and interfaces that simplify the process of creating applications powered by large language models (LLMs) and chat models. LangChain makes it easy to manage interactions with language models, provides prompt templates, and provides memory components to manage and manipulate previous chat messages and incorporate them into chains.

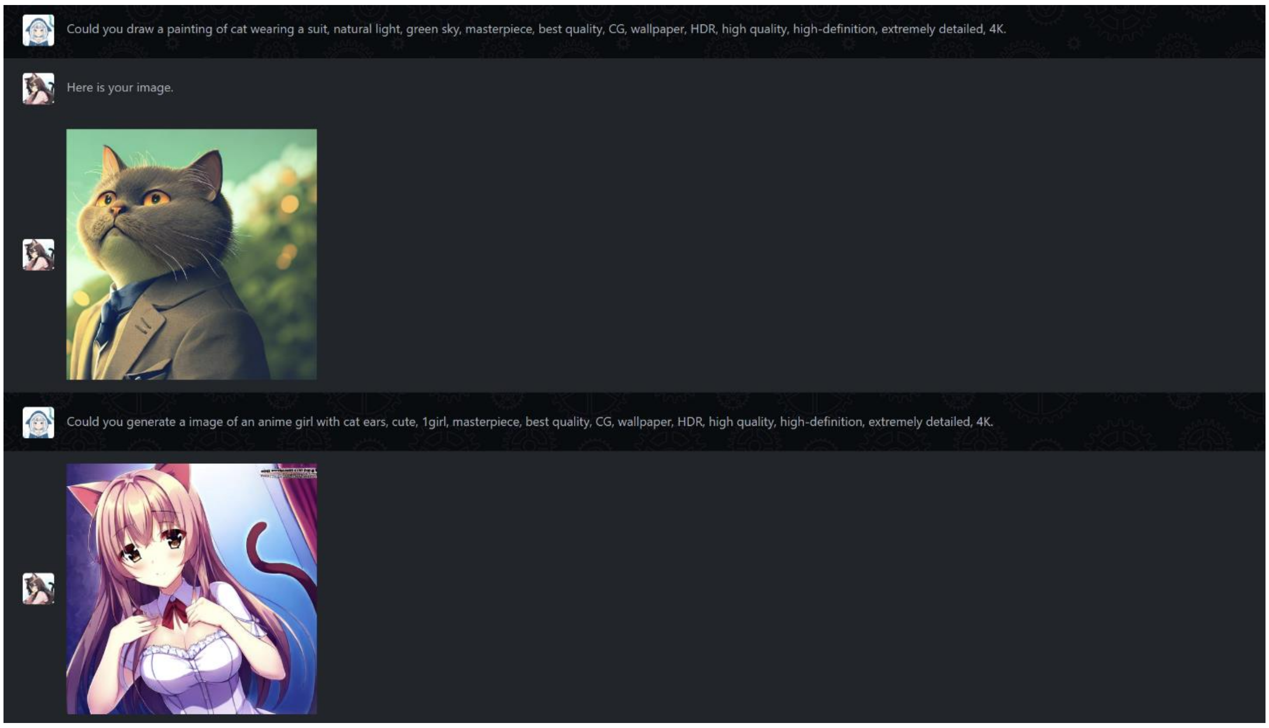
What is the content of my AI homework3?

The content of my AI homework3 is to solve the handwriting recognition problem on MNIST dataset using machine learning methods given in this course, and use the macro f1-score to evaluate the methods. The maximum number of epochs should be no more than 2000. The report should clearly state all the operators and parameters used in each method, and compare the performance of the methods in terms of solution quality, efficiency, and running time. The programming language should be C or C++. The due date is 2023/06/06. Plagiarism is prohibited.

What is MNIST?

The MNIST database is a large database of handwritten digits that is commonly used for training various image processing systems and for training and testing in the field of machine learning. It consists of 28x28 grayscale images of hand-written digits (0-9), with a training set of 60,000 examples and a test set of 10,000 examples. It is often used as a baseline for testing image processing systems and applying a Convolutional Neural Network (CNN) on the MNIST dataset is a popular way to learn about and demonstrate the capabilities of CNNs for image classification tasks.

Fig. 1. Chat History



Could you draw a painting of cat wearing a suit, natural light, green sky, masterpiece, best quality, CG, wallpaper, HDR, high quality, high-definition, extremely detailed, 4K.

Here is your image.

Could you generate a image of an anime girl with cat ears, cute, 1girl, masterpiece, best quality, CG, wallpaper, HDR, high quality, high-definition, extremely detailed, 4K.

Fig. 2. Chat History

Abstract—**This document is a report for term project of lecture - Introduction of Artificial Intelligence by prof. Liaw, Rung-Tzuo. We use the LangChain module in Python to build a GPT-based chatbot that can utilize various tools.**

I. Introduction

Following the release of ChatGPT, many companies are now seeking to utilize Large Language Models (LLMs). We, the individuals without financial resources, strive to adopt the perspective of a startup and endeavor to develop applications at a low cost. In our observation, ChatGPT has four disadvantages for now (April 13, 2023) –

1. It can not search on the Internet.
2. It can not use Tools (like APIs or functions in [1]).
3. It did not have custom database.

It has privacy concerns.

To address these disadvantages, we conducted extensive research and eventually discovered the solution, which led us to the LangChain framework.

So, what is LangChain? LangChain is a framework for developing applications powered by language models. It enables applications that are:

1.Data-aware: connect a language model to other sources of data

2.Agentic: Allow a language model to interact with its environment

As such, the LangChain framework is designed with the objective in mind to enable those types of applications. For example, common use cases include: autonomous agents, personal assistants, answering questions using sources, analyzing structured data, evaluation, summarization, and so on.

Regarding the modules within LangChain, these modules are the core abstractions which we view as the building blocks of any LLM-powered application. For each module LangChain provides standard, extendable interfaces. LangChain also provides external integrations and even end-to-end implementations for off-the-shelf use. The modules are models, prompts, memory, indexes, chains, agents, callbacks.

So, let's have a quick walkthrough about the structrure of building an end-to-end language model application with LangChain using some example modules. The most basic building block of LangChain is calling an LLM on some input. In other word, it can get predictions from a language model, but you are not sending user input directly to the LLM. Instead, you are probably taking user input and constructing a prompt, and then sending that to the LLM. So, that will be the second step, define the prompt template. Next up, we are going to select an agent. Agent use a language model to determine which actions to take and in what order, they are given access to tools, and they repeatedly choose a tool, run the tool, and observe the output until they come up with a final answer. In our project, we will primary focus on the custom tools design and the comprehensive design of the application encompassing both the front-end and back-end Detailed discussions regarding these aspects will be presented in the subsequent report.

II. Related Work

For a chatbot system, we require a front-end for the user interface and input, as well as a back-end for the agent to handle input and generate responses.

A. *Front-end*

For the front-end, we used Next.js, a React.js-based framework, to develop our website. We chose Next.js because it aligns with our familiarity, and as the front-end is not our main focus in this project, we will not delve deeper into it.

B. *Back-end*

For the back-end, we used Google Colab, a cloud server system that provides free Tesla T4 GPU for users. One of our agent's tools, text-to-image, requires GPU computation, making Google Colab the ideal choice for us.

C. *Connecting Front-end and Back-end*

To connect Colab and the front-end, we used the flask-ngrok[2] module to host both the front-end and back-end. This setup allowed us to transform our back-end agent into an API endpoint, where it receives text input and responds with text output. Images can be encoded into base64 format, which allows them to be represented as text.

D. *Writing custom tools for agnet*

In LangChain, a tool is a function that takes text as input and outputs an observation, which is also a text. There are some tools created originally by LangChain. However, LangChain also allows us to customize our own tools for the agent. We involved five tools:

1) Search tool: The search tool is a built-in feature of LangChain. When the agent determines that it requires a tool, it can utilize the search tool to obtain an observation.

2) Text-to-image tool: The text-to-image tool allows the agent to generate AI art when prompted to paint an image. The agent will generate a prompt and utilize the text-to-image model API to create the AI art. Our approach involves utilizing text-to-image models available on Hugging Face. Specifically, we have chosen stable-diffusion-2-1[3] as our preferred model in this tool, considering it to be the most symbolic text-to-image model.

3) Text-to-anime-liked-image tool: The reason we gave the agent another text-to-image tool is that we wondered if the agent truly knows how to choose the appropriate tool. We gave the agent the hakurei/waifu-diffusion[4] model on Hugging Face and telled it that this model is useful when asked to generate anime-like images.

4) Search-memory tool: We stored our AI Homework 3 in a vector database and instructed the agent that it can search for information about AI Homework 3 when asked. The tools will perform a semantic search and return the relevant data from the vector database.

5) Chat tool: We encountered a problem when integrating the agent with tool selection and chatting. Our unsatisfactory solution was to treat chatting as a tool. The agent would decide whether to chat and call the chatGPT API to obtain a response.

III. Methodology

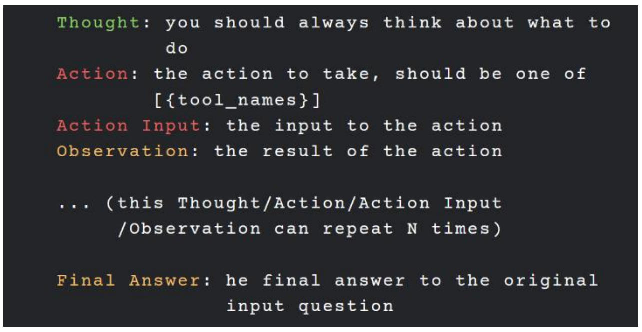
In our project proposal, we stated our intention to fine-tune LLMs. And if this approach proves unsuccessful, our alternative plan is to integrate AI tools into a platform. However, the reason for fine-tuning an LLM is to enable it to incorporate our own data and even utilize tools like an agent(This idea is inspired by AgentGPT[5]). After conducting thorough research and acquiring knowledge, we have discovered that fine-tuning is not a feasible way. Instead, prompting is all we need.

A. *In-context Learning*

According to [6], in-context learning is a mysterious emergent behavior in large language models (LLMs) where the LLM performs a task just by conditioning on prompts, without optimizing any parameters.

In LangChain, this phenomenon is used to teach agent how to utilize tools, similar to [1]. According to its documentation[7], the 'zero-shot-react-description,' which is used in our application, use ReAct framework [8] to teach.

Here is the prompt template:

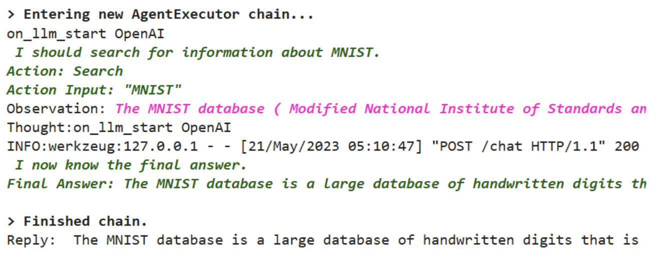


```
Thought: you should always think about what to
              do
Action: the action to take, should be one of
             [{tool_names}]
Action Input: the input to the action
Observation: the result of the action

... (this Thought/Action/Action Input
     /Observation can repeat N times)

Final Answer: he final answer to the original
                   input question
```

Fig. 3. Prompt Template.

The agent will replace the white text, and LangChain will parse the output to call the function tool.

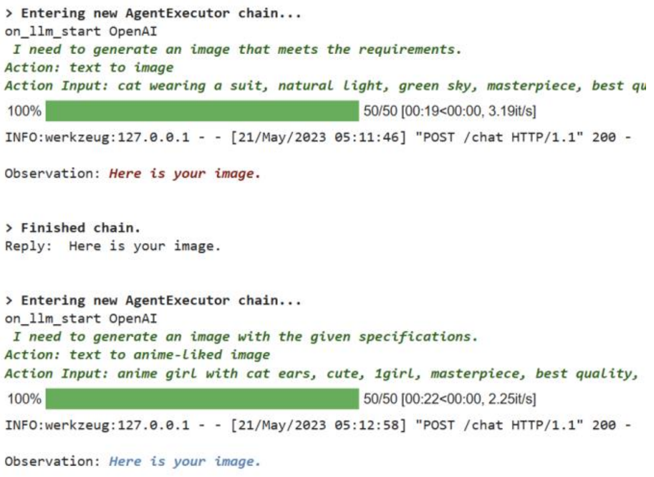
Here is the back-end logs:



```
> Entering new AgentExecutor chain...
on_llm_start OpenAI
 I should search for information about MNIST.
Action: Search
Action Input: "MNIST"
Observation: The MNIST database ( Modified National Institute of Standards an
Thought:on_llm_start OpenAI
INFO:werkzeug:127.0.0.1 - - [21/May/2023 05:10:47] "POST /chat HTTP/1.1" 200
 I now know the final answer.
Final Answer: The MNIST database is a large database of handwritten digits th

> Finished chain.
Reply:  The MNIST database is a large database of handwritten digits that is
```

Fig. 4. Search tool back-end log

In this scenario, we can see the selected agent, which is zero-shot-react-description agent, will utilize the prompt format to execute a chain of thought and generate a response. When the user input "What is MNIST?", the agent's thought process prompt is triggered, leading it to the realization that searching for information about MNIST is the best course of action. Therefore, it utilizes the action input "MNIST" and chooses the search tool to gather relevant information as the action to take. After retrieving the results of the search action, the agent carefully observes the answer and determines its correspondence to the original question. Upon concluding that it has obtained the final answer, the agent formulates the thought "I now know the final answer." and proceeds to deliver the response: "The MNIST database is a large database of...".

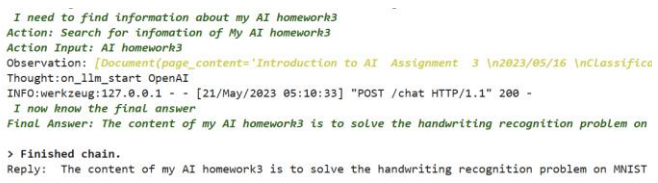


```
> Entering new AgentExecutor chain...
on_llm_start OpenAI
 I need to generate an image that meets the requirements.
Action: text to image
Action Input: cat wearing a suit, natural light, green sky, masterpiece, best qu
100% 50/50 [00:19<00:00, 3.19it/s]
INFO:werkzeug:127.0.0.1 - - [21/May/2023 05:11:46] "POST /chat HTTP/1.1" 200 -

Observation: Here is your image.


> Finished chain.
Reply:  Here is your image.


> Entering new AgentExecutor chain...
on_llm_start OpenAI
 I need to generate an image with the given specifications.
Action: text to anime-Liked image
Action Input: anime girl with cat ears, cute, 1girl, masterpiece, best quality,
100% 50/50 [00:22<00:00, 2.25it/s]
INFO:werkzeug:127.0.0.1 - - [21/May/2023 05:12:58] "POST /chat HTTP/1.1" 200 -

Observation: Here is your image.
```

Fig. 5. Text-to-image tool back-end log

In the first case mentioned above, our user input "Could you draw a painting of cat wearing a suit, natural light, green sky masterpiece, best quality, CG, wallpaper, HDR, high quality, extremely detailed, 4K." and the agent gave the thought "I need to generate an image that meets the requirements.", so it takes the action "text to image", and receive the final answer. In the second case, user input "Could you generate a image of an anime girl with cat ears, cut, masterpiece, best quality, CG, wallpaper, HDR, high quality, extremely detailed, 4K", the agent return the thought "I need to generate an image with the given specifications.", Furthermore, the agent recognizes the requirement for an specific anime-themed image, so it choose the action "text to anime-liked image" tool rather than text-to-image tool.

Fig. 6. Search-memory tool back-end log



```
 I need to find information about my AI homework3
Action: Search for infomation of My AI homework3
Action Input: AI homework3
Observation: [Document(page_content='Introduction to AI  Assignment  3 \n2023/05/16 \nClassifica
Thought:on_llm_start OpenAI
INFO:werkzeug:127.0.0.1 - - [21/May/2023 05:10:33] "POST /chat HTTP/1.1" 200 -
 I now know the final answer
Final Answer: The content of my AI homework3 is to solve the handwriting recognition problem on

> Finished chain.
Reply:  The content of my AI homework3 is to solve the handwriting recognition problem on MNIST
```

Final, we ask "What is the content of my AI homework3?". Since we have already stored our AI Homework 3 in a vector database and instructed the agent that it can search for information specifically related to AI Homework 3 when we make inquiries, so it choose the search-memory tool and subsequently returns the final answer based on that retrieved information.

B. *Word Embeddings and Semantic Search*

If we want ChatGPT to incorporate our own data, we can simply provide it as prompts. However, prompting have some problems… First of all, GPT has a token limit, which means it may not be possible to provide the entire dataset as a prompt if the data is too large.

Secondly, if we separate the data and prompt it to GPT, it might adversely affect its performance. The prompts my become too messy, making it difficult for model to discern the main focus.

Furthermore, if we consider GPT as a company's chatbot and a user wants to ask questions about the company's products, but the user does not have access to the company's data.

To address these issues, a popular approach is to utilize word embeddings and semantic search[9].

1) Word Embeddings: We encode our data into a vector database, where each vector represents the relations between tokens or sentences.

2) Semantic Search: After receiving user input, we search for the k-nearest data in the vector database, and then prompt it to GPT.

IV. Results

The results is in Fig.1. and Fig.2., and here is the explanation for each text input:

1. The user texts the agent with a greeting, and the agent responds normally.

2. The user asks the agent, "What is LangChain?" Unlike when we asked ChatGPT before, the agent utilizes the search tool to observe the results and provide the user with an answer.

3. The user queries the agent for the content of 'AI homework 3,' which simulates custom data for a company. The agent searches its database and provides a reply. This function is similar to chatPDF[10]

4. The user asks the agent, "What is MNIST?", and the agent utilizes the search tool for answer.

5. The user asks the agent for an image, and the agent utilizes the text-to-image tool to generate an image and response it to user.

6. The user asks the agent for an anime-liked image, and the agent utilizes the text-to-anime-liked-image tool to generate an image and response it to user.

V. Conclusions

In this paper, we focus on extending the functionality of ChatGPT for our own use. Our goal is to provide a cost-effective solution that allows users to utilize it directly. To achieve this, we discovered the LangChain framework and utilized it for the application development of LLM (Language Model). We incorporated customized tools and models into the framework and ultimately completed the development of the application called ToolGPT.

VI. Experience and Future Work

A. *Experience*

1) 羅心義: The process of developing this project has been a valuable and enriching experience for me. Throughout this project, I gained hands-on experience in building an effective chatbot system while also fostering strong collaboration with my teammates. In order to solve the problem by developing an AI application, a topic I initially had little understanding of, I delved into extensive research to acquire knowledge about Machine Learning. My journey started with the aspiration to train models from scratch, such as Ranbow Deep Q Neworks or Generative Adversarial Networks(GANs). As I progressed, I explored into the process of fine-tuning existing models like GPT or Diffusion. Eventually, this exploration led me to discover the LangChain framework. During the development process, I found that the most challenging part was establishing an effective communication approach among team members. We held numerous meetings throughout the semester for both homework and project discussions. Our main focus was to ensure that each team member was enthusiastic about the project and to foster a shared understanding among everyone involved. All in all, this project has been a challenge for us, and I am delighted that we have successfully completed it.

2) 尤奕翔: This is my first time working on an AI-related project, and throughout the implementation process, I have learned a lot of new knowledge. Simultaneously, the design and prospects of the project goals have sparked much reflection and raised many questions. I believe this experience will serve as a cornerstone in my journey of learning in the field of AI.There is still so much more for me to explore and learn. And yeah, with that said, I am looking forward to it.

3)黃力宏: This kind of project is new to me in many aspects. First, I have taken only required subject so far, so I certainly have not experienced such team work, which usually appears in elective subject before third year. Second, unlike any other class I have taken, we have to learn everything without being taught so the biggest problem for me in the project is self -learning. Thanks to all my teammates, we shared learning outcomes, which considerably decreased the time we tried to figure out any knowledge that possibly should have. Again, I'm fortunate to have this opportunity for doing such project before our special subject.

4)羅奕晟: In this project, I have learned a lot of fascinating concepts and encountered numerous challenges. I am thrilled to have been part of a team project and successfully achieved our envisioned goals. I also hope to have more opportunities in the future to engage with these highly challenging projects. I believe that AI will play a crucial role in the future trends, and I aspire to explore and delve deeper into the vast and profound field of AI, finding intriguing topics for further learning and application. I am truly grateful to our teacher for providing us with this opportunity for our first attempt.

Future Work

About privacy concern of ChatGPT, we did not delve into this topic extensively during this project. However, it is possible to address this concern by replacing the GPT model with other LLMs such as Alpaca or Vicuna.

We involved three types of tools in this project: text-to-image, search, and search-from-memory. However, due to time constraints, there were some promising tools that we were unable to include. One of them is the Human-as-tool, which enables the agent to ask for help when getting stuck. This concept is intriguing as it prevents agents from looping themselves indefinitely or failing to achieve their goals. Instead, they have the capability to seek advice or receive commands to halt the process. Another tool is the Python REPL (Read-Eval-Print Loop) tool, which accepts Python scripts as input, allowing the agent to write and execute Python code. This tool enhances the potential of the agent.

References

[1] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, Thomas Scialom, "Toolformer: Language Models Can Teach Themselves to Use Tools," arXiv:2302.04761 [cs.CL].

[2] Unknown, "flask-ngrok." [Online]. Available: https://pypi.org/project/flask-ngrok/

[3] StabilityAI, "stabilityai/stable-diffusion-2-1." [Online]. Available: https://huggingface.co/stabilityai/stable-diffusion-2-1

[4] Hakurei, "hakurei/waifu-diffusion." [Online]. Available: https://huggingface.co/hakurei/waifu-diffusion

[5] Reworkd, "AgentGPT", [Online]. Available: https://github.com/reworkd/AgentGPT

[6] Sang Michael Xie and Sewon Min, "How does in-context learning work? A framework for understanding the differences from traditional supervised learning." [Online]. Available: http://ai.stanford.edu/blog/understanding-incontext/

[7] Unknown, "LangChain." [Online]. Available: https://python.langchain.com/en/latest/index.html

[8] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, Yuan Cao, "ReAct: Synergizing Reasoning and Acting in Language Models," arXiv:2210.03629 [cs.CL].

[9] Hugging Face, "Text embeddings & semantic search." [Online]. Available: https://www.youtube.com/watch?v=OATCgQtNX2o

[10] Unknown, "chatPDF." [Online]. Available: https://www.chatpdf.com/

[11]