UNIVERSITÄT ZU LÜBECK

Using AutoGPT for Information Retrieval Agents

Nutzung von AutoGPT für Informations-Recherche Agenten

Masterarbeit

verfasst am
Institut für Informationssysteme

im Rahmen des Studiengangs
Informatik
der Universität zu Lübeck

vorgelegt von
Jakob Horbank

ausgegeben und betreut von
Prof. Dr. Ralf Möller

Lübeck, den 15. April 2024

IM FOCUS DAS LEBEN

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Jakob Horbank

Zusammenfassung

Es geht darum, der Welt »Hallo« zu sagen.

Abstract

It is about saying “hello” to the world.

Contents

1

Introduction

This an introduction like not other. Information retrieval hard and stuff. Would be nice to have a chatbot that answer natural language questions and gives sources from research database and even web.

1.1 Contributions of this Thesis

Hopefully the above.

1.2 Related Work

There are different attempts at creating LLM outputs with sources. Perplexity AI hosts a question answering service, that gives source from websites.

Open Source LLM Agents

- AutoGPT
- babyagi

Information Retrieval Applications

In a variety of application contexts, the answers of an assistent have to be correct. This is espciacally true in reasearch contexts. A model that hallucinates isn't feasable in this case. But while hallucination can be reducing with fine-tuning, it can not be competely eliminated. Perplexity AI is an online service that leverages a language model to provide a search that generates an answer from different sources in the internet. The content of the answer is then linked to the found sources, so the user can see and verify the result.

As this is a propierty closed source product accessible through a web interface, this is not a useful to create our own research assistant. The provided API simply hosts different LLMs and allows for prompting them. There is no possibilty of of fine-tuning.

1.3 Structure of this Thesis

I do this then that and then that.

2

Backgrounds

Different branches of research were used to build upon in this work. These topics and how their are connected to my work are explained in more detail in this section.

2.1 Information Retrieval

Handling large databases filled with information of different is a common problem. There has been extensive reasearch on database architectures and indexing algorithms.

2.2 Agents

Although the notion of agent is not new, it recently gained attention in combination with the rise of generative language models.

2.3 Large Language Models

Natural language processing is a long studied research area of computer science. In recent years delevopments have significantly sped up, with the introduction of deep learning into NLP. The transformer archticture has been the basis for all advancements in recent years.

Transformer

Until 2017, the dominating strategy to train models for language tasks revolved around recurrent structures. Every sentence token was represented as a hidden state that resultet from all the previous tokens. While this approach has a reasonable motivation, the sequential nature contraints computation speed. There is no way to compute the recurrent architectures in parallel.

The transformer model [1] solely relys on these attention mechanisms. In particular, they employ self-attention and multi-headed self-attention layers

Attention mechanism allow learning dependencies between tokens in sentences without regard to their distance. Their non-sequential nature allow for massive parallelization.

Self-Attention is an attention mechanism that computes relations between different positions of the same sequence with goal of finding a representation of the sequence.

Like a typical sequence modeling architecture, a transformer consists of an enconder and a decoder. The encoder has two parts that are stacked on each other multiple times. The first part is a mulit-head self-attention block, the second part is a classic fully connected feed foward block.

BERT

The BERT model family consists of encoder only transfomers. They are trained to generating rich embeddengs of tokens for use in a range of different downstream tasks.

GPT

GPT models are decoder only transformer models. They excel at text generation tasks. they are trained to predict the next token of a sequence.

Instruction tuned Language Models

Language models trained to predict the next token of a sequence. While alot of knowledge is captured in the weights of the model, most information in the internet is not formatted in conversatonal style. Because of this, langauge models need to be prompted in a specific way to be effective. The prompt has to be written in a way that the continuation of it yields the desired output. This is not optimal for human users, as they would rather write in a conversational style. Instruction tuned models solve that problem.

Instruct models are fine-tuned version of language models. Capturing the intent of the user is a key challenge for language models. This process is called *alignment.* popular approach to the alignment problem is reinforcement learning with human feedback (RLHF). Handcrafted prompts are used to fine-tune GPT-3. The outputs of the model are collected into a set and ranked by humans. This set is then used to train a reward model. With this reward model, the language model is further fine-tuned. The resulting model is called *InstructGPT* and performs better than the baseline GPT-3 model.

Large language models are trained to predict the next token of a sequence, not to follow the instruction of the user. This leads to some unwanted results, such as toxic, harmful answers or fabricated information that is not true.

LLM Agents

AutoGPT is an open source project that tries to leverage LLMs to function as an agent controller. GPT models are extended with different modules to creat an agent that receives a goal and then acts towards reaching it. To reach the goal AutoGPT plans, has a memory, and reflects on past actions.

AutoGPT is an example implementation of an agent. The project tries to become a framework to build agents with different architectures. AutoGPT employs the key components of an agent outlined in [2]. The profile module allows the agent to assume different roles, such as expert, teacher or coder. Profile are LLM specific as each model responds best for different prompting styles. Memory is implemented through a database.

Other open source llm agent systems

- miniagi miniagi
- babyagi miniagi

3

Analysis of AutoGPT for Information Retrieval Tasks

AutoGPT is an open source project that tries to 'make GPT fully autonomous'. GPT langauge models are used to control an agent that works towards reaching a stated goal. The project contains a core general purpose agent with a predefined set of abilities. Addidtionaly a baseline sdk is beeing developed to build custom agents.

3.1 LLM Agent Backgrounds

The impressive capabilities of large language models have lead to researchers implementing them in agent systems.

3.2 Architecture Overview

The AutoGPT agent is modeled after the classic agent architecture. After a goal given to the agent, it creates a task that has to be completed. Then it completes steps that make up that task. In each step an action is run.

AutoGPT is divided into four modules. The *brain* is the main module that controls the agent. In AutoGPT this is realized by prompting the language model in a structured way. Using the chat system prompt, the language model is prompted to answer a structred format. The format is shown in listing 1. Different techniques are implemented in this structure. The language model is forced not only to plan the next step, but also to explain the choice for the chosen step and to add self-criticism. An extra output for the human user is also returned. The second part of the answer is the actual next action with the needed arguments. The action make up the second module of the agent. In this module the abilities of the agent are defined. These can be file operations, database queries or web search functionalities. The third module is the *memory.* Memories are modeled after humans which have short and long-term memory. Short term memory can be implemented as an in-memory list messages to the language model. Long-term memory needs a persistent store such as a database. A popular option for language models are vector databases that work with embeddings.

```
Reply only in json with the following format:

{
    \"thoughts\": {
        \"text\":  \"thoughts\",
        \"reasoning\": \"reasoning behind thoughts\",
        \"plan\": \"- short bulleted\\n- list that conveys\\n- long-term plan\",
        \"criticism\": \"constructive self-criticism\",
        \"speak\": \"thoughts summary to say to user\",
    },
    \"ability\": {
        \"name\": \"ability name\",
        \"args\": {
            \"arg1\": \"value1", etc...
        }
    }
}
```

Listing 1: AutoGPT System Prompt

Currently, AutoGPT is only compatible with OpenAI models. Switching to a different model is not that hard, because only the API format would have to be changed. The challenge is to switch between different prompting styles, as every model needs to be prompted differently. For example OpenAI models benefit from profile sentences like ”You are an expert in computer science”, while Anthropics Claude does not...

3.3 Information Retrieval Capabilities

The AutoGPT Agent has different abilities that can be uitilzed for information retrieval. Notably, it is able to search the web and operate on the file system.

The web search is implemented by a two step process. First a search API like duckduckgo is called to get a list of relevant pages. Then the page contents are scraped with a headless browser. It is possible to read and write to text files. Other document types are processed by basic text extraction tools to get the plain text.

For longer files such as scientific journals the extracted fulltext is too long for the language model. The AutoGPT agent has no ability to chunk the text into smaller chunk or store it in a database. This is a limitation that needs to be addressed for information retrieval tasks over a research database repository.

Having a vectorestore would enable techniques such as retrieval augmented generation. The agent would get a prompt which a question over the RDR and choose an action to start a semantic search over the vectorstore. The result of the search are the chunks that are semantically closest to the question. These chunks can then be inclued as context for the LLM prompt to generate an answer.

The default agent has a tendency towards searching the web for information. We want an agent that prioritizes information that is present in the research repository. This needs to be addressed in the prompting techniques of the agent.

4

Retrieval Augmentated Generation Agent

Large language models are excellent at generating text in a variety of styles. Different approaches try to embed information sources into the generated text. One appreach is to fine-tune the language model on a dataset that contains the information. This can work, but often there is not enough data to shift the model weights enough. A more promising approach is retrieval augmented generation. Before generating an answer to a prompt, a vectorestore is searched for relevant information. The retrieved information is then included in the prompt.

4.1 Retrieval Augmented Generation

- what is a vectorestore compared to other databases
- how are documents stored and queried
- how is the information included in the prompt

4.2 Extending AutoGPT with retrieval capailities

- Custom agent Loop
- Custom abilities
- Further guidances for the agent

5

Creating IR Agent Benchmarking Challenges

The evaluation of large language model agents is a difficult task, as evaluating LLMs themselves presents a challenge. There are different approaches to evaluate systems built around language models. Subjective evaluation is based on human feedback. As LLM systems are generally made to serve humans this is an important part of evaluation. On the other hand, quantiative metrics that can be computed are used for objective evaluation. Different metrics are used for different tasks. Another importat method are benchmarks. Benchmarks are a set of tasks or an environment that the agent is to move in.

5.1 Existing IR Agent Benchmarks

As the space of possible agent domains is large, lots of different benchmarks were proposed. Simulation enviroments like

Although lots of benchmark for LLM applications have been proposed, there are few benchmarks that are designed to test the information retrieval capabilities of an agent. There are some benchmarks that are used to evaluate information retrieval in general and in a diverse domains like ”cite”.

5.2 The AutoGPT Benchmarking System

To evaluate AutoGPT and other agent systems that implment the agent protocol, the AutoGPT project has implemented its own benchmarking system. The system constists of a set of tasks that the agent has to complete. The tasks are designed to test different aspects of the agent, and are divided into different topics. Some tasks depend on the previous succesful completion of other tasks. A task consists of an input prompt and a expected output. The ouput is defined by certain words that should be contained.

- Level based system
- Dependencies

5.3 Custom Benchmarks for local IR over journals

- How i benchmark the steps in IR

6

Results

7

Conclusion

Saying hello world is quite easy.

Bibliography

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is All you Need. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[2] Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., et al. *A Survey on Large Language Model based Autonomous Agents*. Aug. 2023. DOI: 10.48550/ARXIV.2308.11432. arXiv: 2308.11432 [cs.AI].