Projektbericht:
Nutzung von AutoGPT für Informations-Recherche Agenten

Jakob Horbank

29. Februar 2024

Übersicht

Ich habe nach meiner Recherche ein Proof-of-Concept Retrieval Augmentation System (RAG) gebaut. Da das System funkionierte, habe ich dann die RAG Idee in einen eigenen AutoGPT Agenten eingebaut. Dabei habe ich mich gegen eine Modifikation des Standard AutoGPT Agenten entschieden, da dieser kontinuierlich von den Entwicklern verändert wird und viele Fähigkeiten besitzt die über den Bereich dieser Arbeit hinausgehen.

AutoGPT

Ich will hier kurz auf das Projekt AutoGPT an sich eingehen, da sich der Zustand des Projektes über die Zeit verändert hat. Dabei hat sich die Aktivität um AutoGPT im Laufe meiner Arbeit merklich verringert. Im ersten Monat meiner Arbeit wurde ein Community Hackathon veranstaltet und eine zweistellige Millionensumme an Fördergeldern für AutoGPT eingesammelt. Über die Monate wurden konstruktive Forenbeiträge immer seltener und viel mehr wurde gefragt ob das Projekt noch am Leben ist. Einige Ideen wurden bereits von großen Unter-nehmen wie OpenAI[1] übernommen, und die Weiterentwicklung scheint immer langsamer voranzugehen.

Daten

Zu Beginn habe ich mit einem Datensatz bestehend aus annotierten arabischen Schriften gearbeitet. Da dieser Datensatz aber eher für Klassifikationsaufgaben geeignet ist, verwendete ich jetzt die *Studies in Manuscript Cultues* Serie [2]. Die Artikel enthalten sehr spezifisches Wissen und eignen sich daher gut für Information Retrieval Aufgaben in Expertendomänen.

Experimente mit Retrieval Augmented Generation

In meinen Recherchen zu den Themen Information Retrieval, Agenten und LLMs bin ich immer wieder auf Retrieval Augmented Generation gestoßen (RAG). Daher habe ich mich dazu entschieden einen RAG-Agent zu bauen.

Ein RAG System besteht aus folgenden Schritten Die Informationen, die für die Informationsrecherche relevant sind werden in eine Datenbank indiziert. Hierfür wird wird meist eine Vektordatenbank verwendet, die es ermöglicht Dokumente mit ihren Embeddings nach semantischer Ähnlichkeit zu Clustern. Der Nutzer stellt nun eine Anfrage an das RAG System und bevor eine Antwort generiert wird, wird die Vektordatenbank zu semantisch relevanten Dokumenten angefragt. Das Sprachmodell erhält die relevanten Dokumente als Kontext in der Anfrage, und wird instruiert Information aus diesem Kontext zu verwenden.

[1] https://platform.openai.com/docs/assistants/overview
[2] https://www.csmc.uni-hamburg.de/publications/smc.html

Um die Idee besser zu verstehen habe zuerst eine statische RAG Pipeline mit einem lokalen LLM gebaut. Da dieses Proof-of-Concept System funktionierte, entschied ich mich dazu mit diesem Ansatz und einem AutoGPT Agenten weiterzuarbeiten.

Standard AutoGPT Agent für IR

AutoGPT beeinhaltet einen Standardagenten, ein Framework um eigene Agenten zu entwickeln und ein Benchmarking System das sich mit eigenen Tests erweitern lässt. Der Standardagent bestitzt Fähigkeiten die für Information Retrieval nützlich sind, wie Web-Search und simple Datei-operationen. Jedoch fällt auf dass der Agent nicht für lokale IR optimiert ist, und dazu tendiert Onlinesuchen durchzuführen. Da der Agent eine große Codebase besitzt die täglich verändert wird, habe es ich es für einfacher gehalten das Framework für einen eigenen Agenten zu verwenden, als den Standardagenten anzupassen.

RAG Agent für IR

Ich habe mit dem Framework einen RAG Agenten für IR gebaut. Der Agent hat entsprechende Fähigkeiten zum indizieren von Dokumenten, Anfragen der Datenbank und Generierung einer Antwort erhalten. Das als Datensatz verwendete PDF wird in Klartext konviertiert und dann in kleinere Chunks aufgeteilt, die ein Embedding Model verarbeiten kann. Nach der Indizierung der Chunks in die Vektordatenbank, kann der Agent auf diese als Informationsquelle zugreifen. Über die mitgelieferte AutoGPT GUI kann der Agent wie ein Chatbot angefragt werden, während eines Chatverlaufs ist es ebenso möglich die Anfrage zu verfeinern. In jedem Schritt wählt der Agent eine Fähigkeit aus um der Beantwortung der Nutzeranfrage näher zu kommen und gibt eine Begründung sowie Selbstkritik aus. Was noch fehlt ist eine Rückverlinkung der Dokumente, die als Kontext für die Generation der Antwort verwendet wurden.

Da LLMs jedoch nicht immer eine determistische Antwort geben und auch nicht immer das gewünschte Antwortformat einhalten funktioniert dieser Ablauf nicht immer. Es kann passieren, dass bei der gleichen Anfrage verschiedene Aktionen ausgeführt werden.

Evaluation

Die Evaluation eines LLM-Agenten ist schwierig, da die Entscheidungen nondeterministisch generiert werden. Im Benchmarking System von AutoGPT werden die Tests einfach mehrfach wiederholt in der Hoffnung, das es oft genug funktioniert. Ich habe vor bekannte Evaluationsdatensätze für Information Retrieval zu ver-wenden um die grundlegenden Fähigkeiten des IR Agenten zu evaluieren. Anschließend werde ich versuchen mit einem LLM spezifische

Testbeispiele aus dem genannten Datensatz zu generieren um den Agenten auf spezifischem Wissen zu testen.