18.05.2023

Как работает AutoGPT внутри?

Введение

Существует множество видео на YouTube и статей в блогах о AutoGPT, но ни одно из них не заглядывает под капот и не рассматривает сам код. Поэтому я изучил код и предоставлю краткий обзор того, как он работает, а также некоторые интересные моменты кода, которые я нашел неожиданными. Это не претендует на полное описание кода, так как это было бы слишком длинным. Я рассмотрю только интересные моменты.

Общий поток выполнения

При запуске приложения пользователю предлагается ввести цель для "AGI". После получения этой цели, AGI пытается разработать план и выполнить его.

Вот высокоуровневый поток выполнения:

Термины

> Full message history (Полная история сообщений) - Вся "беседа" до сих пор. Мы заставляем его общаться сам с собой.
>
> System prompt (Системный запрос) - Сообщает ChatGPT о его возможностях и том, как он должен отвечать.
>
> Triggering prompt (Инициирующий запрос) - Цель, которую пользователь хочет достичь.
>
> Vector memory (Векторная память) - Долгосрочная память, представленная в виде векторных вложений.

- Процесс начинается с вызова API ChatGPT с полной историей сообщений, системным запросом, инициирующим запросом и векторной памятью.
- Затем ChatGPT отвечает в формате JSON.
- План действий выводится на печать.
- Учитывается обратная связь пользователя. Пользователь может одобрить действие, предоставить критику, отклонить и завершить или запросить собственную критику.
- Если предоставлена обратная связь, процесс не выполняет никаких действий и возвращается к шагу 1. Это механизм безопасности.
- Если действие одобрено, выполняется запланированная команда, например, поиск в Интернете или написание кода.
- Результат затем добавляется в историю сообщений как "системное".
- Наконец, проверяется условие завершения. Если выбран выход, процесс завершается. В противном случае он возвращается к шагу 1.

System Prompt

Системный запрос - это постоянная строка, которая служит вводным текстом к чат-сообщению. Он состоит из следующих частей.

Я сохранил эту строку в файле и прикрепил его здесь.

Хотя системное время не является частью прикрепленного текстового документа, оно также передается в системный запрос.

Ограничения, Ресурсы и Оценка Производительности

Следующая строка предоставляется без изменений модели чата. Я думаю, что этот аспект сам по себе ясен.

Ограничения:
Ограничение в ~4000 слов для краткосрочной памяти. Ваша краткосрочная память ограничена, поэтому немедле
Если вы не уверены, как ранее сделали что-то или хотите вспомнить прошлые события, мышление о похожих соб
Нет помощи от пользователя.
Используйте исключительно команды, перечисленные в двойных кавычках, например, "имя команды".

Ресурсы:
Доступ в Интернет для поиска и сбора информации.
Управление долговременной памятью.
Агенты, основанные на GPT-3.5, для делегирования простых задач.
Вывод в файл.

Оценка производительности:
Постоянно рассматривайте и анализируйте свои действия, чтобы убедиться, что вы работаете наилучшим образо
Конструктивно самокритикуйте свое поведение в целом постоянно.
Отражайте прошлые решения и стратегии, чтобы усовершенствовать свой подход.
Каждая команда имеет свою стоимость, поэтому будьте умными и эффективными. Старайтесь выполнять задачи
Записывайте весь код в файл.

Команды

AutoGPT взаимодействует с миром с помощью команд, которые могут быть чем угодно, начиная от поиска в Интернете и заканчивая созданием подагентов.

ChatGPT отвечает командой и аргументами, например: {"command": "write_to_file", "arguments": {"file_name":"foo.txt", "contents": "Hello world"}}

Этот JSON-ответ разбирается, и вызывается связанная с этой командой python-функция.

Commands:
1. analyze_code: Analyze Code, args: "code": "<full_code_string>"
2. read_audio_from_file: Convert Audio to text, args: "filename": "<filename>"
3. execute_python_file: Execute Python File, args: "filename": "<filename>"
...
21. Task Complete (Shutdown): "task_complete", args: "reason": "<reason>"

Категории:

```
command_categories = [
    "autogpt.commands.analyze_code",
    "autogpt.commands.audio_text",
    "autogpt.commands.execute_code",
    "autogpt.commands.file_operations",
    "autogpt.commands.git_operations",
    "autogpt.commands.google_search",
    "autogpt.commands.image_gen",
    "autogpt.commands.improve_code",
    "autogpt.commands.twitter",
    "autogpt.commands.web_selenium",
    "autogpt.commands.write_tests",
    "autogpt.app",
    "autogpt.commands.task_statuses",
]
```

Большинство команд предсказуемы, но есть несколько необычных:

Команда выполнения кода может работать в контейнере Docker.
Команда "google" для поиска использует DuckDuckGo.
Команда "google_official_search" использует Google.

Формат ответа

Мы просим ChatGPT возвращать ответ в формате, который может быть разобран с помощью JSON parse. Однако он не всегда возвращает правильно структурированный JSON, поэтому в коде есть вспомогательные функции для разбора почти правильных JSON.

You should only respond in JSON format as described below
Response Format:

```
{
   "thoughts": {
      "text": "thought",
      "reasoning": "reasoning",
      "plan": "- short bulleted\n- list that conveys\n- long-term plan",
      "criticism": "constructive self-criticism",
      "speak": "thoughts summary to say to user"
   },
   "command": {
      "name": "command name",
      "args": {
         "arg name": "value"
      }
   }
}
```

Ensure the response can be parsed by Python json.loads

Вот пример JSON из цели AutoGPT "создать успешную учетную запись в Twitter":

```
{
   "thoughts": {
      "text": "Now that we have a plan, let's optimize our profile and bio. We should start by updating our profile picture and
      "reasoning": "Optimizing our profile and bio is a crucial first step in building a successful Twitter account. It sets the tor
      "plan": "- Update profile picture and header image
            - Write a clear and concise bio
            - Include relevant keywords and hashtags",
      "criticism": "We need to make sure that our profile and bio accurately reflect who we are and what we do. We should a
      "speak": "Let's optimize our profile and bio. We should update our profile picture and header image to something eye-c
   },
   "command": {
      "name": "write_to_file",
      "args": {
         "filename": "profile_optimization.txt",
         "text": "- Update profile picture and header image
               - Write a clear and concise bio
               - Include relevant keywords and hashtags"
      }
   }
}
```

Self feedback

Самооценка является необязательным шагом, который можно включить, нажав клавишу "s", когда AutoGPT предлагает ответить "да" или "нет".

AutoGPT берет сгенерированные мысли и отправляет их без контекста в новый вызов API, запрашивая обратную связь.

Обратная связь просто добавляется к истории чата.

Обратная связь_мысли - это мысли из вышеуказанного JSON-ответа.

feedback_prompt = f"Below is a message from me, an AI Agent, assuming the role of {ai_role}. whilst keeping knowledge o

```
add_to_chat_history([
    {"role": "user", "content": feedback_prompt + feedback_thoughts}
])
```

Полная история сообщений

Полная история сообщений - это полезная нагрузка запроса, которая отправляется в API ChatGPT. Она выглядит так, как показано на приложенном изображении.

Векторная память

Все более популярным способом обхода ограничений размера контекста является использование векторных баз данных.

Для тех, кто не знаком, семантическая модель вложения (обычно SBERT, но в автогпт используется модель OpenAI ada) используется для генерации вектора для каждой мысли.

Семантически похожие тексты имеют более высокое скалярное произведение, чем непохожие тексты.

Вот шаги для использования векторных баз данных в качестве долгосрочной памяти:

- Запросить векторную базу данных с вложением текущей мысли.
- Получить 5 наиболее похожих мыслей.
- Сделать новый API-запрос для суммирования 5 мыслей в более короткий текст.
- Подать эту сводку вместе с другой информацией.

Динамическое резюме

Динамическое резюме также используется для отслеживания выполненных действий на данный момент.

```
# Initial memory necessary to avoid hallucination
self.summary_memory = "I was created."

new_events = [
    {"event": "entered the kitchen."},
    {"event": "found a scrawled note with the number 7"}
]
update_running_summary(new_events)
# Returns: "This reminds you of these events from your past: \nI entered the kitchen and found a scrawled note saying 7."
```

Уведомления о бюджете

AutoGPT также указывает модели ChatGPT поторопиться и завершить работу, чтобы уложиться в бюджет.

```
system_message = (
    f"Your remaining API budget is ${remaining_budget:.3f}"
    + (
        " BUDGET EXCEEDED! SHUT DOWN!\n\n"
        if remaining_budget == 0
        else " Budget very nearly exceeded! Shut down gracefully!\n\n"
```

```
        if remaining_budget < 0.005
        else " Budget nearly exceeded. Finish up.\n\n"
        if remaining_budget < 0.01
        else "\n\n"
    )
)
```

Под-агенты

AutoGPT может создавать подагентов с определенным именем, у которых есть свой собственный контекст и назначенная цель. Это кажется ещё одной стратегией для преодоления ограничений размера контекста.

На момент написания этой статьи, агенты работают не в отдельных подпроцессах, а в синхронизированных функциях. Коммуникация происходит с помощью строк на естественном языке и, на практике, аналогична вызову функции. Агент вызывается с аргументами и возвращает результаты. Каждый агент ведет свою собственную историю чата.

Принятие решений о создании, перечислении, отправке сообщений и завершении агентов осуществляется с помощью обсуждаемых выше команд.