Relatório 2 - Prática: Python: Criando um ReAct Agent do Zero (I)

Isadora Stéfany Rezende Remigio Mesquita

Descrição da atividade

O objetivo dessa atividade foi desenvolver um agente de IA do zero, sem fazer uso de frameworks de orquestração (como LangChain e LanghGraph). Dessa forma, permitindo a consolidação do nosso conhecimento adquirido no card 1 de forma prática.

Vídeo: Python: Criando um ReAct Agent do Zero

Neste vídeo, foi possível aprender de forma prática como criar um agente de ReACT do zero com python puro, sem a utilização de estruturas prontas. A explicação é muito clara e detalhada, facilitando a compreensão de como funciona o padrão de reação em agentes de IA.

O conceito de agente foi explicado como um loop automatizado que interage com um LLM, buscando respostas e aprimorando perguntas. Isso otimiza o processo de obtenção de informações. O padrão ReAct é introduzido como uma maneira intuitiva de projetar agentes, utilizando um ciclo de **observação, ação e pensamento**. Esse padrão facilita a interação com o mundo real.

Por meio do loop, o agente é capaz de observar o contexto, entender, decidir por uma ação (como utilizar ou não uma ferramenta), pensar novamente e posteriormente, chegar a sua melhor conclusão. Por meio desse processo, o agente é cada vez mais capaz de obter informações precisas, específicas e conseguir trazer uma resposta mais assertiva.

No caso do agente construído no tutorial, a ferramenta construída foi uma pesquisa simulada na Wikipédia para auxiliar no processo de busca de informações.

A construção do nosso agente é feita por meio de uma classe, e ao inicializar essa classe, definimos atributos como client (acesso ao nosso modelo de LLM), system (prompt utilizado para construir o agente e como ele deve agir) e métodos. O método call é responsável por anexar mensagens ao histórico do agente, permitindo que ele tenha um registro da conversa com o usuário. O loop de pensamento, ação e observação, como mencionado antes, permite que o agente interaja de forma dinâmica às mensagens recebidas. Além disso, foi criado o método execute, o qual permite anexar mensagens e retornar um resultados. Essa funcionalidade é essencial para o aprendizado do modelo, uma vez que ele está entendendo o que foi solicitado, como agir, e o que concluir com base em toda a iteração.

No vídeo, o objetivo do agente é calcular a massa de planetas no sistema solar e realizar cálculos com esses valores. Para isso, duas tools foram construídas: get_planet_massa e calculate. Essas ferramentas são informadas no prompt, e o agente tem conhecimento do acesso a elas. A primeira representa uma busca simulada no Wikipédia, e a segunda representa uma calculadora.

Dessa forma, é possível solicitar para o agente qual é a massa de um planeta, como a Terra, e qual o seu valor multiplicado por 5 ou através de outra operação matemática.

Com relação a minhas aplicações, confesso que fiquei bastante empolgada com a ideia de construir uma busca na web. A primeira aplicação foi muito semelhante a do tutorial, mas focada em um assistente de informações sobre seriados e livros, em especial, Game of Thrones, que sou bastante fã. Acredito que seria bastante interessante se a Omelete tivesse um desse para que seus acompanhantes pudessem usufruir.

A minha segunda aplicação foi um pouco mais complexa, e precisei utilizar a ferramenta ChatGPT para construir a minha tool, que é de fato uma busca na web. Por não conhecer muito

a respeito das tecnologias utilizadas na construção de sites, eu não sabia muito bem como construí-la. Porém, após essa atividade, além de exercitar o meu conhecimento a respeito de agentes de IA, pude conhecer um pouco melhor a respeito de divs e cabeçalhos utilizados na construção dessas aplicações.

Bom, a ideia era que o agente fosse capaz de realizar uma consulta real na web para atender a necessidade de busca do usuário e com base no perfil dele. Para isso, o usuário insere suas informações para cadastro, que deveriam ser armazenadas em um DB, e em seguida realiza a sua consulta. O Agente, com acesso a todas essas informações, deve realizar buscas que satisfaçam a necessidade do usuário. Com receio de informações deturpadas, passei uma lista de sites mais confiáveis. Além disso, gostaria que ele fosse capaz de recomendar clínicas próximas a localidade da pessoa que está realizando a busca, se necessário, porém não consegui evoluir a aplicação tanto assim por enquanto.

Ao finalizar essa aplicação, observo como um ponto de melhoria a criação do DB Fake. Não seria algo tão difícil de ser feito, então acredito que poderia ter criado como uma ferramenta a parte, assim como fiz na primeira aplicação. Isso poderia evitar que a pessoa sempre digitasse as suas informações de cadastro para compor esse DB Fake, e em vez disso, apenas informar o seu nome e CPF ou id de cadastro pudesse fazê-la acessar com mais facilidade. Enfim, melhorias futuras.

Com este card, pude compreender melhor o funcionamento de um agente ReAct, desde sua estrutura até sua execução. Aprendi como o prompt desempenha um papel essencial na construção do agente, orientando seu comportamento e tomada de decisão. Além disso, ficou claro como é possível implementá-lo de forma eficiente utilizando apenas Python.

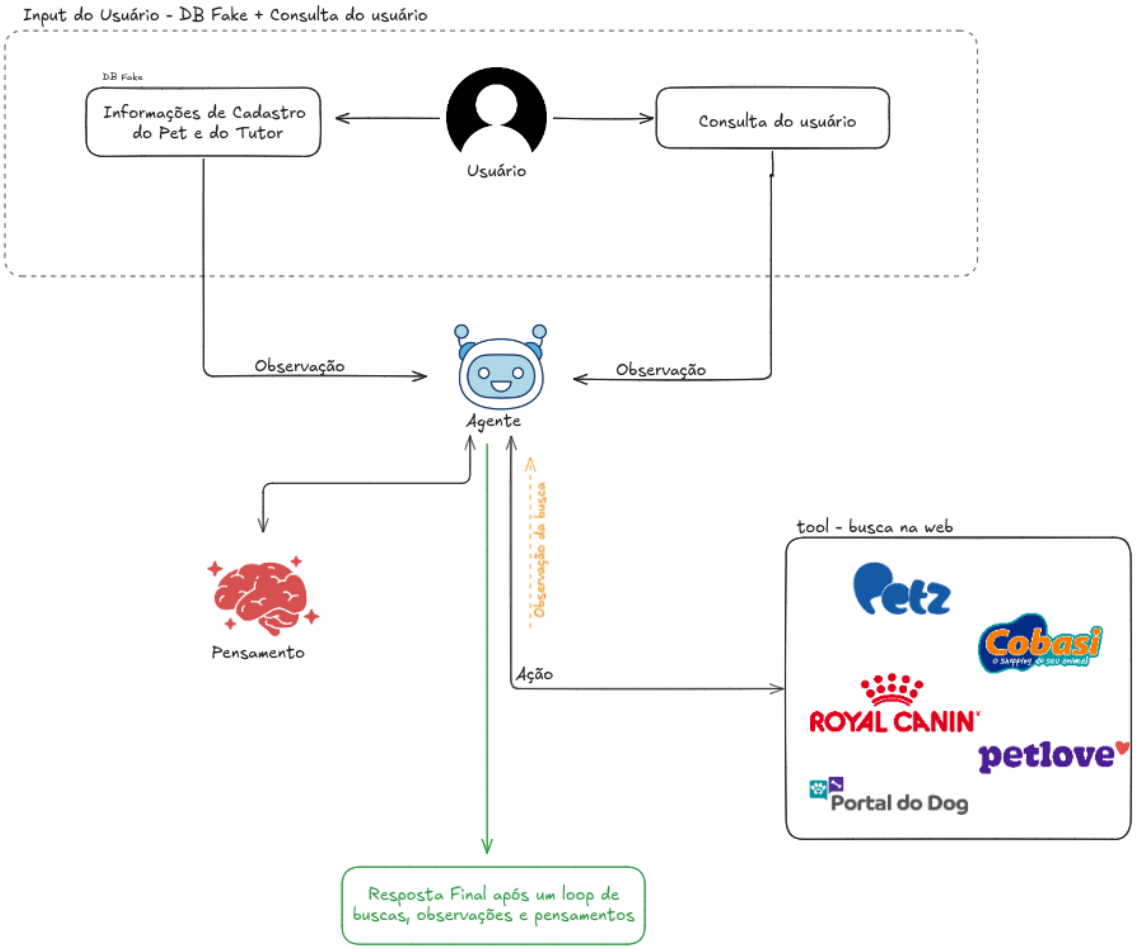

Figura 1: Fluxo do Agente Assistente de Tutores de Pet.

Dificuldades

Não percebi dificuldades no curso em si. Mas confesso que foi um pouco confuso construir a tool de acesso a sites para a minha segunda aplicação.

Conclusões

Ficou muito claro como um Agente ReAct funciona e como construí-lo. Além disso, eu não conhecia a plataforma Groq. Fiquei muito animada e achei interessante a facilidade que nos permite acessar LLMs. Ademais, aprendi um pouco mais a respeito de estrutura de sites. Vi brevemente ao longo da graduação, mas confesso que entendo muito pouco. Nessa parte, contei com a ajuda de ferramentas como o ChatGPT para me auxiliar na construção dessa ferramenta que, a princípio, era bem nebulosa como poderia ser feita. Nessa implementação, conheci um pouco mais a respeito de divs e cabeçalhos utilizados na construção dessas aplicações.

Referências

Python: Criando um ReAct Agent do Zero