**Como usar a lib langchain para cria um agente - criando o seu chatgpt personalizado - Vídeo 3**

**Transcrito por** [**TurboScribe.ai**](https://turboscribe.ai/pt/?ref=docx_export_upsell)**.** [**Atualize para Ilimitado**](https://turboscribe.ai/pt/subscribed?ref=docx_export_upsell) **para remover esta mensagem.**

Quer entender como os agentes da Biblioteca Langchain funcionam? Fica comigo nesse vídeo. Oi, pessoal, aqui é o Rafa. No vídeo de hoje, eu vou mostrar para vocês os agentes, que é o terceiro componente da solução da Biblioteca Langchain que estava faltando aqui na série inicial.

A ideia é fazer vários tutoriais de Langchain. E, só fazendo uma pequena recapitulação, no primeiro vídeo, eu mostrei o que era a biblioteca e, basicamente, ela tem três grandes componentes. Elas têm os LLMs, elas têm as chains e têm os agentes.

LLMs é como se fosse o cérebro, certo? É o motor por trás de todas as soluções. É o chat GPT, é o Gemini, é o Llama. São esses LLMs que vão ser utilizados para a inteligência da nossa aplicação, para lidar com processos ou tomadas de decisão.

Nesse vídeo, ainda, eu mostrei aquela primeira ideia da chain, da corrente da biblioteca, onde você cria prompts com alguns passos pré-definidos, onde você tem variáveis de input definidas e, ao output, você pode encadear isso. Então, por exemplo, tem um prompt que vai me dar as principais cidades turísticas de um país e, para cada uma desses outputs, eu posso encadear e me der locais para estar nessa cidade. Então, por exemplo, se eu estivesse pensando em criar uma função de turismo, eu sempre vou receber o país no meu prompt e, como output, eu tenho as cinco cidades principais desse país.

E, para cada uma dessas cidades, eu posso enviar para um outro prompt e esse outro prompt vai me dar os principais pontos turísticos de cada uma das cidades que o primeiro prompt gerou. E, assim, eu posso encadear quantos processos eu quiser. E a primeira e única grande diferença entre as chains e os agentes é que, no caso das chains, esse processo é estático, ele é hard-coded.

O que quer dizer isso? Que sempre que eu executar ele, ele vai esperar que eu mande o nome de um país e ele vai desencadear todo esse encadeamento de processos que eu, como programador, criei. Então, eu já tenho que saber de antemão o que eu quero utilizar numa chain, certo? Se eu usar as chains, as cadeias. Processos estáticos, processos que não mudam, processos que eu não vou ficar inteirando muito.

A chain é o ideal, onde você vai juntar o LLM em prompt e vai conseguir criar a sua solução. Os agentes são um passo adiante dessa questão da chain, onde a gente cria soluções que os agentes conseguem escolher o que fazer. Então, por exemplo, eu vou criar um agente médico, vamos supor assim, um bot médico, e ele recebe informações de um paciente e, dado as informações do paciente, ele tem que escolher qual é a ação a tomar.

Eu peço medicamento, eu peço para esse cara ser internado, eu dou alta para ele. São processos diferentes que ele pode tomar. Cada um desses processos, a gente vai chamar de ferramentas, de tools, que são as possibilidades que o LLM pode agir e pegar.

Quem vai decidir o que usar e em que ordem usar é o agente. Essa é a grande e principal diferença. E na aplicação que a gente vai fazer hoje, vou botar aqui na mão na massa, criar um agente aqui em Python, a gente vai criar um agente bem simples, sem nenhum propósito, de novo, igual o vídeo de HAG, para mostrar para vocês como funciona a tecnologia.

É conceitual, não é uma solução. Eu vou criar um agente, eu vou criar duas tools diferentes. Uma que busca links de vídeos no YouTube e a outra que busca informações do Google Trends.

Eu vou mostrar para vocês como eu crio essas tools, como eu testo elas, como eu crio um agente que faz isso, e depois eu vou interagir um pouco com o meu agente. Se eu perguntar para ele um link de YouTube, eu espero que ele me devolva esse link de YouTube. Se eu perguntar para ele alguma coisa do Google Trends, eu espero que ele me dê as informações do Google Trends.

Certo? Então, vamos lá colocar a mão no código e descobrir como é que funciona o agente. Pessoal, estamos aqui já, então, no ambiente de desenvolvimento. E da mesma maneira que a gente utilizou os embeddings da OpenAI na última solução, a gente vai utilizá-las aqui também.

Então, aqui lembrando que sempre que na pasta env eu já tenho a minha chave lá da OpenAI, então eu vou ter que mandar ela para o sistema. Eu vou usar a biblioteca .env, que já faz isso para mim. E aqui, simplesmente, instancia essa função e ela vai pegar todas as chaves que estiverem aqui, as variáveis de ambiente que estiverem aqui nesse arquivo .env e vão mandar para o meu ambiente de execução.

Certo? O agente, basicamente, vai ter as tools, certo? Ele vai precisar de um objeto de prompt para rodar. E, além disso, ele vai precisar, é claro, do LLM. Então, a gente vai criar esses três objetos.

A gente vai, depois, juntar as tools junto com o prompt e o LLM. E esses três caras juntos são, de fato, o meu agente, certo? Vou colocar em inglês, porque, geralmente, toda a literatura ainda é em inglês, então vamos manter o inglês aqui. Só que isso aqui é uma representação, uma classe vai representar o meu agente e eu preciso de alguma coisa para executar.

Então, no caso, o W, a tecla lang change, a gente precisa de um executor. Um executor. Então, são esses três objetos que a gente vai precisar aqui.

Vamos, então, criar, primeiro, as tools que a gente vai usar. Na verdade, vamos já adiantar um pouco. Vamos criar, primeiro, o LLM, que a gente já sabe que a gente vai usar o do chat.gt mesmo.

Então, vamos aqui importar ele. Então, LLM igual OpenAI. Aqui, de novo, só lembrando como a minha chave está na variável de ambiente, não vou precisar passar de novo.

Vou usar o GPT 3.5 Turbo, já que é uma versão mais barata. E a ideia, depois, é mostrar para vocês as soluções com ferramentas open source, onde não vai ter nenhum custo. Aqui, se eu estou acertando a temperatura igual a zero, eu não quero que ele seja nada criativo, tá? Eu quero que ele use o mínimo possível e alucine o mínimo possível, né? Então, por isso que eu quero que ele use as informações que vai buscar aqui nas tools, tá? Então, já tenho o meu LLM aqui estanciado.

Vamos criar, agora, as nossas tools. Eu falei para vocês que a gente vai utilizar duas, basicamente, né? Uma que é o YouTube, o buscador de YouTube, e a outra que é uma do GoTrends. Todas essas soluções, elas estão na parte de community do Lungchain.

Community são pessoas como eu e você que criam soluções e buscam divulgar essas soluções junto com outras pessoas para utilizá-las. Então, qualquer um de nós pode criar uma outra solução e disponibilizar para outras pessoas utilizarem. E é interessante, esse é um ponto que muita gente fala, ah, agora acabou a carreira, e ah, não vai ter mais carreira de programador, de dev.

Não, gente, só vai mudar. Quem você acha que criou todas essas coisas? São as pessoas que estão antenadas no mercado, são as pessoas que já sabem que isso é balela, que só mudou o mercado. Então, quem criou esse YouTube Search Tool? Foi alguém como eu e você, que entendeu que deveria ser legal ter uma ferramenta para buscar vídeos no YouTube a partir de LLMs de maneira programática.

Então, está aqui na biblioteca, criaram, fizeram um deploy. O time da Lungchain deixou válido esse deploy, então a gente pode utilizar ele oficialmente. E aqui é muito simples.

Para criar essa aqui, eu só vou criar uma variável, eu vou chamar, deixa eu pensar, YouTube Tool, e ela vai ser igual a esse cara instanciado. Então, com esse cara aqui, o LLM vai utilizar esse objeto para entrar no YouTube e pesquisar vídeos a partir de uma palavra. É isso que essa ferramenta faz, certo? Dado uma URL, ela busca esses links.

Então, vou agora criar a minha outra ferramenta, que é a do Google Trends, que está na mesma biblioteca, certo? Vou pegar aqui, eu volto a copiar, ela está dentro de Tools, só que ela tem uma subpassa, que é o Google Trends, certo? Se eu quiser saber quanto uma palavra está sendo usada ou algo desse gênero. Então, aqui, Google Trends, Query RAM. E eu vou precisar de uma outra dependência também, que está em Utilities.

E essa dependência é o Google Trends API Wrapper. Preciso desses dois objetos para fazer o Google Trends funcionar. Vamos aqui embaixo, então, Google Trends.

Vou instanciar esse cara e vou passar um API underline Wrapper, igual a esse. Mas, Rafa, de onde é que você tirou essas coisas? Como é que eu descobri essas coisas? A resposta é a resposta menos sexy do mundo, a resposta que talvez quem esteja começando na área está cansado de ouvir, que é leia a documentação. Então, se a gente vê aqui na documentação, por exemplo, temos aqui o YouTube falando o que eu preciso para fazer isso funcionar.

Então, está aqui na aba de Community, como se importa. Ele explica o que é esse pacote, ele busca vídeo no YouTube e você não precisa ter uma conta no Google, ele dá um bypass, um limite de pesquisa. Você não precisa dar a API do YouTube para utilizar essa solução deles, porque é muito legal isso.

De novo, alguém como eu e você criou isso daqui. A gente importa dessa maneira, eu sei que são as dependências que tem que instalar, que eu já instalei, a gente instancia ele e simplesmente passa esse exemplo aqui. Aliás, vamos dar uma rodada nesse cara antes? Eu sei que esse não é o propósito.

Vou só colocar aqui. Vamos dar um print, só que ao invés de Lex Friedman, vamos mudar isso aqui para LangChain, que é no vídeo sobre LangChain. Aqui, vamos rodar agora o nosso código.

Tem um de sintaxe aqui, né? Corrigimos. Ah, gente, pelo amor de Deus, faltou o G aqui. Agora sim.

Sem esse G, tudo certo. Todas as nossas dependências estão corretas. Vamos rodá-lo e ver o que ele vai devolver aqui para a gente dessa ferramenta.

Vamos rodá-lo aqui agora. Aqui, temos dois vídeos. Vamos ver se esses dois vídeos tratam sobre LangChain.

Então, temos aqui um vídeo sobre LangChain. Então, a API não é necessariamente uma API, né? É uma biblioteca capaz de, a partir de uma palavra, ela ir no YouTube e buscar vídeos relacionados a um determinado tema. Bem interessante essa API aqui.

Agora, vamos... Não, API não, gente. Estou com a API na cabeça. Vamos testar.

O Google Trends é a mesma coisa. Se a gente vir aqui na documentação dele, ele fala que provavelmente essa empresa aqui, essa API, e a gente tem que ser... Já estou aqui nela. Você tem que ter uma conta, você vai ter uma chave, assim como você tem na OpenAI, e ele fala que você tem que ser uma chave de ambiente, ser API underline key.

Eu posso pegar essa variável de ambiente, usar a mesma estratégia que eu usei aqui, colocá-la aqui dentro desse .env, para expandir a coisa que eu já fiz, e eu vou poder usar o Google Trends aqui também. Então, vamos ver o funcionamento dele aqui. Basicamente, lá que eu descobri que eu preciso desse whopper e desse query run, esse é o objeto que, de fato, vai executar.

Aqui, por exemplo, a palavra water. Ele vai falar water, o número médio dessa palavra procurada, as tendências, tem histórico. Ele me dá uma string gigante com todas as informações.

Provavelmente eu já dei uma string aqui porque a gente vai utilizar isso no nosso LLM. Essa é a parte interessante de já ter uma biblioteca do Longchain com essa biblioteca poderosa para trabalhar com LLMs. E aqui, gente, não é nada demais.

E-mail, vai ter que validar com o telefone. Você vai ser logado e vai para a sua área de usuário, e lá você vai ter só copiar e colar a sua chave API. Sem grandes elucubrações aqui.

O terceiro passo que a gente vai precisar... Bom, aqui a gente já tem o nosso LLM. A gente vai precisar de um prompt. O prompt que a gente vai utilizar... A gente vai pegar um prompt próprio já da OpenAI, que, lembra, a Longchain é uma open source.

Ela não é da OpenAI. Você pode usar vários LLMs. Inclusive, a minha ideia para os vídeos futuros é trazer outros LLMs diferentes da OpenAI para vocês terem outras opções.

Mas aqui, para facilitar nossa vida, a OpenAI já criou um template para funções, para agentes que usam o LLM da OpenAI. Esse template aqui. OpenAI functions agent, certo? Está o nome dele aqui.

E eu vou usar a função rib do Longchain ativo justamente para importar esse cara. Aí já importa ele na raiz e usa o template. Não precisa criar o nosso.

Mas dá bem para a gente criar um template próprio e ir melhorando esse template. Tenho aqui, então, o meu prompt. Esse pool é de ir pegando mesmo.

Eu quero aquele modelo da OpenAI. Esse é provavelmente o dev que desenvolveu. OpenAI.functions-agent, certo? Então, a gente já tem aqui praticamente tudo que a gente precisa.

A gente precisa juntar essas tools. A gente sempre vai juntar todas as tools que a gente quiser que o agente seja capaz de fazer em uma lista de tools. Então, simplesmente pego esses objetos aqui, copio e colo.

Se eu tiver mais coisas que o meu agente quer fazer, pode fazer, eu tenho que substanciar eles aqui e colocar em uma lista. No exemplo que eu dei sobre o médico, por exemplo, aqui poderia ser perdida de exames, perdida de radiografia. Aí pode colocar outra coisa, perdida de alta, perdidos de um remédio X. Faça o teste e tal.

São coisas que o agente pode fazer. Toda tool é uma coisa que o agente pode fazer. A gente simplesmente substancia elas aqui e coloca em uma lista chamada de tools.

E agora eu tenho já os três componentes que eu preciso. Tenho as tools, tenho o prompt para usar e tenho o LLM. Então, agora a gente vai juntar isso tudo em um agente.

E vamos utilizar aqui os agentes. E vou pegar aqui a função createOpenAIFunctionsAgent. A partir dessa função, vou aqui no meu LLM, vou substanciar ele.

E aqui eu simplesmente passo o LLM que eu criei. Depois eu vou passar a minha lista de tools e em seguida vou passar o meu prompt. Certo? Como eu havia falado no começo, um agente nada mais é do que a junção de um LLM, das tools e do prompt.

O prompt é para ele receber as informações, algum template. Aqui, no caso, já está usando o da OpenAI, que aqui do lado já está dizendo como ele funciona e tudo mais. Então, por isso que a gente não está perdendo tempo.

Eu já mostrei para vocês como se cria um prompt customizado no primeiro vídeo dessa série. Estamos usando esse aqui por conta disso. Em seguida, a gente está utilizando o LLM, substanciou ele aqui.

Então, passando isso tudo para o nosso agente. Mantendo o inglês aqui. E como eu falei para vocês, uma coisa é quem vai, de fato, executar isso.

A gente precisa de uma ferramenta para executar os nossos agentes. Então, aqui, vamos importar o AgentExecutor. Esse objeto é o responsável por executar esse agente.

Então, de novo, algo bem simples aqui. Eu vou criar esse objeto. Executor.

E a gente vai falar que o agente desse cara é o agente. As tools são as tools, certo? Porque a gente definiu essa lista aqui em cima. Eu vou deixar ele verboso, porque eu quero que vocês vejam o que ele está pensando.

Vou deixar ele true aqui. Ele vai colocar vários logs. É bem legal para a gente ver o que ele está escolhendo.

Qual o processo que ele está escolhendo para a gente. E pronto. Acabamos de criar o nosso primeiro agente.

Tudo o que a gente precisa tem aqui. Recapitulando, temos as tools, que é o que ele pode fazer. O prompt é como ele recebe informação e devolve a informação.

O LLM, que seria entre aspas o cérebro. O que vai orquestrar essa solução. E temos aqui o agente que junta tudo.

Então, essa aqui é uma definição abstrata. Para a gente ter algo que execute, a gente precisa do agent executer. E agora, vamos começar com os testes.

Vamos começar com um teste bem simples. Vamos dar aqui um invoke. E a gente passa como se estivesse passando um dicionário.

Input. E vou começar como se fosse uma pessoa. Falando, olá, tudo bem? Vamos supor, por exemplo, que isso fosse um chatbot de atendimento ou algo nessa linha.

Então, é um bot que vai interagir com o ser humano. Opa! O que eu errei aqui? Ah, claro, tem que instanciar ele. Estou passando uma referência.

Errou o bolo. Vamos rodar ele aqui de novo. Lembrando que eu mandei um olá, tudo bem? Olá, tudo bem? Como posso te ajudar? A gente poderia, a partir disso, encadear outras perguntas.

Mas, simplesmente, vou começar do zero. Vou... como... ter memória, né? A gente podia fazer ele ter memória, como se fosse um chatbot, e lembrar do que ele estava interagindo. Agora, eu vou fazer uma pergunta, e eu quero que essa pergunta utilize essa tool ou essa tool.

Então, eu vou fazer algo relacionado ao YouTube, por exemplo. Me dê alguns links de vídeos aqui no YouTube que falam sobre LLMs. Vamos rodar aqui agora.

Aqui, ele começou a executar. Olha só, ele invocou YouTube Search. Olha que legal.

Ele entendeu que ele tinha que vir para cá. Eu não dei nenhuma dica para ele aqui, né? Eu falei um prompt, como um ser humano falaria. Aí, ele me responde aqui.

Aqui estão os links de vídeos do YouTube que falam sobre LLMs. Ele só está aparecendo nessa parte aqui, porque eu deixei o verboso aqui, para mostrar para vocês o que ele está fazendo. Mas, a rigor, se isso aqui fosse falso, ele só mostraria aqui para a gente a resposta.

Vamos lá. Só confirmar. Pegar esse primeiro vídeo aqui.

E... Um vídeo aqui do Google sobre LLMs. Certo? Então, funcionou. Agora, vamos fazer um outro prompt.

E vamos perguntar algo sobre o Google Trends. Deixa eu ver. Qual o valor médio do interesse pela... Vou colocar mais recente.

Pelo termo LLMs. E veja que interessante aqui. Eu não preciso falar para ele, cara, use essa tool do Google Trends.

Use essa outra do YouTube. Não. Ele vai saber interagir com a ordem correta.

Inclusive, se ele não souber, não tiver tool, você tem que criar estratégias para lidar com isso. Se ele vai dizer que não sabe, se ele vai mandar derivar para um ser humano. Aí, é uma arte, como você lida com o caminho não feliz quando você está criando um chatbot.

Mas, vamos ver aqui de novo. Ele entendeu que ele tinha que ir para cá, para o Google Trends, com LLMs. Esses aqui são os parâmetros.

Isso aqui é o que ele devolveu na query. E aqui é a resposta dele. O valor médio de interesse mais recente pelo termo LLM é aproximadamente 50.45, que é o que está aqui.

Interessante. Interesse 350% comparado com as empresas anteriores. Faz sentido, né? Estamos na era dos LLMs.

E aqui foi uma breve explicação para vocês sobre como é que funciona um agente. É um exercício bem inicial, né? A minha ideia é que isso é algo que todo mundo possa entender. Tenho ideias para próximos vídeos.

Já penso em aplicações um pouquinho mais sofisticadas. E agora, vamos fazer as considerações finais. Pessoal, algumas considerações finais em relação à aplicação que a gente viu.

De novo, o primeiro é uma aplicação de brinquedo para vocês entenderem a diferença das ferramentas, entender o que é um prompt, entender que o agente junta os dois. E o principal, que o agente escolhe a ordem e qual ferramenta que ele vai executar, sem a gente precisar dar informação nenhuma para ele. E eu espero que vocês entendam o quanto fantástico e poderoso é isso para nós conseguimos construir ferramentas otimizadas, construir nossos próprios chat GPTs, criar nossas soluções com LLM.

Em casa, se a gente precisa, por exemplo, na empresa, com dados proprietários, a gente pode usar o RAG, a gente pode criar, por exemplo, ferramentas que buscam, fazem queries em SQL, em bancos de dados, e explorem aquela parte de community da Langechain, da biblioteca Langechain, que tem uma infinidade de componentes, de tools já prontos, que é só pegar e usar. E eu acredito que vai ser muito útil no dia a dia de vocês para criar essas pequenas automações e deixar a vida de vocês mais eficiente. Então, esse papo de que a IA vai tirar o emprego do programador, que as pessoas não vai mais aprender tecnologia, balela, gente.

Olha só o quanto de coisas as pessoas estão fazendo. Olha só o quanto de inovação as pessoas estão criando. Quem você acha que criou esse tipo de estrutura? Quem pensou isso? Obviamente foi um programador, foi um ser humano, foi uma pessoa da área de tecnologia.

E se você é um profissional de IA, você tem que ter capacidade de entender onde está o desenvolvimento da tecnologia, onde você vai ser útil, como você pode se sobressair. E é isso que eu quero com essa playlist, que vocês peguem essa playlist e criem as ideias de vocês. E depois que vocês criaram essas ideias, faz o teste aí em casa e bota aqui embaixo para mim o que vocês testaram, manda o link do GitHub, vamos conversar que eu vou ter muito prazer em dialogar com vocês e ajudar vocês nessa jornada de conhecimento.

Se gostou do vídeo, não esquece de curtir, compartilhar e comentar. E até a próxima. Tchau, tchau.

**Transcrito por** [**TurboScribe.ai**](https://turboscribe.ai/pt/?ref=docx_export_upsell)**.** [**Atualize para Ilimitado**](https://turboscribe.ai/pt/subscribed?ref=docx_export_upsell) **para remover esta mensagem.**