IMPLEMENTAÇÃO DE UM FAQ INTELIGENTE PARA EDITAIS ACADÊMICOS COM MODELOS DE LINGUAGEM DE GRANDE ESCALA E TÉCNICAS DE RAG

Matheus Siston Galdino[1] (EG), Mirela Vitoria Domiciano[1] (EG), Ryan Felipe Ferreira Ribeiro[1] (EG), Egon Luiz Muller Junior (PQ)[1]
[1]*Universidade Federal de Itajubá - Campus Itajubá.*

Palavras-chave: Inteligência Artificial. Chatbots. LangChain. Recuperação de Contexto. Embeddings.

Introdução

O uso de inteligências artificiais (IA) tem como uma das principais finalidades a implementação de agentes conversacionais, ou *chatbots,* que utilizam linguagem natural para interagir com usuários (NICOLESCU; TUDORACHE, 2022). Esses *chatbots* são ferramentas efetivas para fornecer respostas automatizadas e precisas a consultas, facilitando a comunicação e o atendimento em diversos contextos (NIRALA et al., 2022). Assim, a motivação do projeto está em auxiliar os servidores da Universidade Federal de Itajubá (UNIFEI) no atendimento de dúvidas relacionadas aos editais lançados pela instituição. Dessa forma, como todas as informações estão especificadas nesses documentos, um agente conversacional pode fornecer respostas de maneira eficiente e automatizada, otimizando o processo de atendimento.

Ademais, o trabalho teve como objetivo a construção de um agente conversacional capaz de responder às perguntas de usuários sobre um edital específico da UNIFEI, funcionando de maneira semelhante a um sistema de perguntas frequentes (FAQ). Para isso, a aplicação foi desenvolvida utilizando como base o *framework* LangChain, que permite a interação e manipulação de uma *Large Language Model* (LLM), mais especificamente o Meta Llama 3 70B. Ainda assim, para gerar embeddings a partir do arquivo PDF do edital, utilizou-se o modelo ada-002 da OpenAI. Além disso, o agente emprega técnicas de *Retrieval-Augmented Generation* (RAG) para realizar consultas ao banco de dados vetorial, ChromaDB, armazenando e consultando estas representações vetoriais de modo eficiente. Desse modo, o agente é capaz de fornecer respostas coerentes e relevantes com base nas informações contidas no documento.

Metodologia

Os LLMs são capazes de interagir de forma natural e responder perguntas sobre diversos assuntos. Contudo, sem o uso de técnicas auxiliares, sua capacidade se limita aos dados utilizados em seu treinamento. Uma dessas técnicas é conhecida como RAG, que combina o poder de métodos matemáticos de comparação de vetores com a capacidade dos LLMs de gerar texto em linguagem natural.

O primeiro passo do processo consiste em carregar os dados que serão utilizados pelo LLM. Esses dados são convertidos para o formato vetorial, o que possibilita sua busca e armazenamento em um banco de vetores, um tipo de banco de dados adaptado e otimizado para operações vetoriais. A seguir, as etapas utilizadas para a elaboração do agente:

1. Extração: Utilizou-se o PyPDFLoader do LangChain para extrair o texto do documento PDF e torná-lo manipulável.

2. Particionamento: Cada vetor armazenado no banco possui um tamanho fixo, que varia conforme o modelo de *embedding* utilizado. Para o modelo ada-002 temos vetores de 1536 dimensões, isso significa que o texto transformado em formato vetorial terá 1536 dimensões de representação. Devido a isso, é necessário dividir o documento em pedaços de texto, garantindo uma maior precisão, uma vez que pedaços muito grandes podem reduzir a eficácia da busca já que teremos as mesmas 1536 posições para representá-lo.

3. Geração de *embeddings* e indexação: Utiliza-se o modelo de *embeddings* para transformar o texto em formato vetorial. Em seguida, esse conjunto é inserido no banco de vetores e um índice é gerado. O objetivo do índice é reduzir a latência, organizando o espaço vetorial de forma que não seja necessário percorrer todos os vetores para encontrar o mais semelhante, sem comprometer significativamente a precisão.

A partir do momento em que a base está disponível e indexada no banco de vetores, é possível realizar operações de busca, também conhecidas como busca de

similaridade para recuperação de contexto. O processo se inicia com a entrada do usuário que, no caso de um sistema de FAQ, ocorre por meio de uma pergunta. Essa pergunta é convertida para o formato vetorial utilizando o mesmo modelo de *embedding* previamente aplicado ao banco de vetores. Com esse vetor, inicia-se o processo de comparação com os vetores armazenados buscando os n pedaços de texto mais similares à consulta do usuário.

Para a métrica de similaridade, optou-se pelo uso de semelhança de cossenos, embora outras métricas como distância euclidiana (L2) e o produto vetorial também possam ser utilizadas. Ao final desse processo, os vetores mais próximos são recuperados e convertidos novamente em um formato textual. Com o contexto relevante recuperado, inicia-se o processo de geração da resposta. Esse processo envolve o uso de um *prompt* no seguinte formato: “ Você é um assistente especializado em responder, dado um contexto, perguntas relacionadas ao edital de transferência interna da Universidade Federal de Itajubá. Baseado no seguinte contexto: {contexto recuperado inserido aqui}, responda a seguinte pergunta: {pergunta inserida aqui} ”

Essa instrução é passada ao LLM que então gera uma resposta com base no contexto recuperado, a qual é devolvida ao usuário. Segue abaixo a estrutura do agente construído para representar visualmente o sistema:

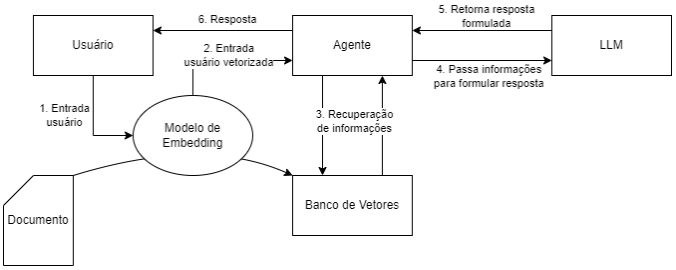

Figura 1 – Estrutura do agente

Para facilitar o acesso ao FAQ optou-se por uma interface seguindo um modelo simples de chat, utilizando o *framework* Streamlit que é capaz de gerar interfaces gráficas básicas usando a linguagem Python. Além disso, todo o ambiente de desenvolvimento seja *back-end* e *front-end* estão rodando em ambiente local utilizando a ferramenta Docker que auxilia no encapsulamento desses ambientes facilitando um posterior deploy da aplicação.

Resultados e discussão

Para validar a solução, criou-se uma base de dados sintética contendo 29 perguntas e suas respectivas respostas, baseadas nas informações do edital Nº 35/2024 (Processo Seletivo Para Transferência Interna, Transferência Facultativa e Portador de Diploma de Curso Superior - Cursos Bacharelado e Licenciatura). Esta base foi desenvolvida usando o modelo GPT-4o, um modelo avançado que oferece suporte abrangente para geração de texto coerente e relevante.

As perguntas foram submetidas ao FAQ inteligente, e os resultados obtidos foram armazenados em um arquivo JSON. Essas respostas foram então avaliadas quanto à precisão e coerência, visando garantir a consistência com as informações do edital. Para essa avaliação, utilizou-se o modelo GPT-4 para comparar o conteúdo da base de dados com o gerado pelo FAQ. Para cada pergunta, foi determinado se a resposta estava correta ou errada e o grau de semelhança com a base de dados.

Indicador	**Valor**
Porcentagem de Acerto	82.75%
Porcentagem de Erro	17.24%
Média de Semelhança	4.30
Motivos de Acerto	
Atende a resposta original	24
Motivos de Erro	
Informação errada	1
Informação incompleta	0
Não encontrou a informação	4

Tabela 1 – Indicadores dos Resultados Obtidos

A interface desenvolvida para acessar o FAQ inclui as mensagens trocadas entre o usuário e o chat, o logotipo da UNIFEI, uma caixa para digitação de perguntas e um botão para limpar as mensagens exibidas na tela. Esta configuração está ilustrada na figura abaixo:

Figura 2 – Interface Gráfica do Chat

Conclusões

O uso de modelos de linguagem em combinação com técnicas de recuperação de contexto podem se tornar uma boa alternativa para automatizar processos como suporte de perguntas e respostas de editais acadêmicos, atingindo uma boa porcentagem de acerto. O desenvolvimento de agentes conversacionais inteligentes, como o apresentado neste trabalho, oferece uma solução eficiente para otimizar o atendimento em instituições acadêmicas, proporcionando respostas rápidas e precisas a consultas baseadas em documentos formais, como editais. A integração LLMs com técnicas de RAG, mostrou-se eficaz na construção de um sistema de FAQ automatizado.

Os resultados obtidos evidenciam que o agente é capaz de fornecer respostas coerentes e relevantes com uma taxa de acerto de 82,75%, demonstrando sua aplicabilidade em contextos que demandam precisão na recuperação de informações. As falhas identificadas, como a não recuperação ou a entrega de informações erradas, representam um desafio a ser aprimorado em futuros trabalhos, que podem explorar melhorias na geração de *embeddings* e na estratégia de recuperação de contexto.

Para a próxima etapa do projeto, planeja-se a validação da solução em um cenário real, utilizando o edital de transferência interna da UNIFEI. Será feito o aprimoramento da interface gráfica para garantir uma experiência de usuário mais intuitiva e eficiente.

Agradecimentos

Agradecemos aos colegas e professores do PET-TEC por promoverem um ambiente de inovação tecnológica que permitiu o desenvolvimento deste trabalho. Nossa gratidão também à UNIFEI e ao FNDE pelo suporte e incentivo, essenciais para a realização deste projeto.

Referências

DOCKER. *Documentation*. Disponível em: https://www.docker.com/ . Acesso em: out. 2024.

LANGCHAIN. *Documentation*. Disponível em: https://python.langchain.com/docs/introduction/. Acesso em: out. 2024

NICOLESCU, Luminița; TUDORACHE, Monica Teodora. *Human-computer interaction in customer service: the experience with AI chatbots—a systematic literature review*. Electronics, v. 11, n. 10, p. 1579, 2022.

NIRALA, Krishna Kumar; SINGH, Nikhil Kumar; PURANI, Vinay Shivshanker. *A survey on providing customer and public administration based services using AI: chatbot*. Multimedia Tools and Applications, v. 81, n. 16, p. 22215-22246, 2022.

OPENAI. *Embedding Models Documentation*. Disponível em: https://platform.openai.com/docs/guides/embeddings/embedding-models . Acesso em: out. 2024.

STREAMLIT. *Documentation*. Disponível em: https://streamlit.io/ . Acesso em: out. 2024.