Relatório Projeto ChatBot Unicamp 2024

O presente documento se dispõe a apresentar os resultados obtidos durante a construção de um ChatBot que responde dúvidas sobre o vestibular Unicamp 2024, bem como explicar detalhes da implementação e como testá-lo.

Modelo

Para a realização do projeto, foram utilizados os conceitos de RAG (Retrieval Augmented Generation). Inicialmente, o conteúdo do site da Unicamp que contém as informações a respeito do vestibular 2024 foi salvo em PDF para ser utilizado como modelo.

O principal desafio da primeira etapa do projeto (lidar com os dados do PDF) era em relação às tabelas presentes no documento. Uma tentativa inicial de converter o documento para um arquivo do formato txt para em seguida fazer a leitura dos dados não se mostrou eficiente, tendo em vista que os dados da tabela eram lidos coluna por coluna, e dessa forma, não havia uma sincronia entre as diferentes colunas de uma mesma linha. A solução foi, então, fazer a leitura diretamente do arquivo PDF, e ao analisar as chunks, verificou-se que as tabelas eram lidas linha a linha, conforme desejado. Para testar se o modelo respondia bem às tabelas, foi feita uma pergunta específica, cuja informação se encontrava em uma das tabelas. A resposta do modelo foi positiva, como pode-se observar na segunda linha do arquivo “Respostas ChatBot.csv” do GitHub.

Logo após a leitura do arquivo, todo conteúdo foi salvo em uma string, que posteriormente foi particionada em chunks de 1024 caracteres cada. Após pesquisas sobre RAG, notou-se que um tamanho de chunk de aproximadamente 500 tokens era o mais adequado para o treinamento do modelo, e com isso, foram feitos testes de particionamento a fim de chegar o mais próximo possível de 500 tokens. Com 1024 caracteres, o histograma para o número de tokens por chunk obtido encontra-se na figura abaixo.

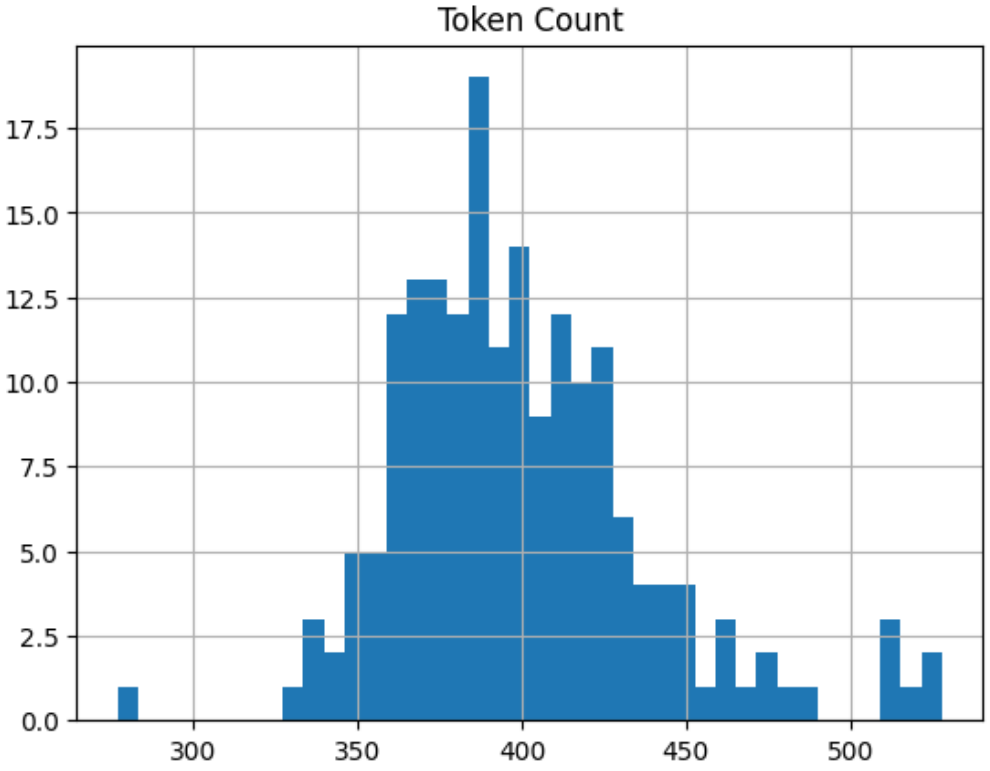

Figura 1 - Número de tokens por chunk

Em seguida, foi criado um vetor database utilizando as chunks e o embedding model da OpenAI. Quando o usuário digita sua pergunta, ela é também vetorizada, e aplica-se uma busca de similaridade para encontrar vetores de valor semântico compatíveis.

Resultados

Para avaliar os resultados, foram organizadas 20 perguntas, com resposta esperada e resposta fornecida pelo modelo. Os resultados do modelo estão presentes na planilha “Respostas ChatBot.csv” do GitHub, que foi utilizada como DataSet. As perguntas foram feitas utilizando, inclusive, linguagem abreviada, que foi perfeitamente compreendida pelo ChatBot.

A partir dos dados dessa planilha, utilizou-se a biblioteca “spacy” para comparar a similaridade semântica entre a resposta esperada e a resposta fornecida através da técnica de “Cosine Similarity”. Os resultados coletados foram utilizados para a construção de um histograma, que encontra-se na figura abaixo.

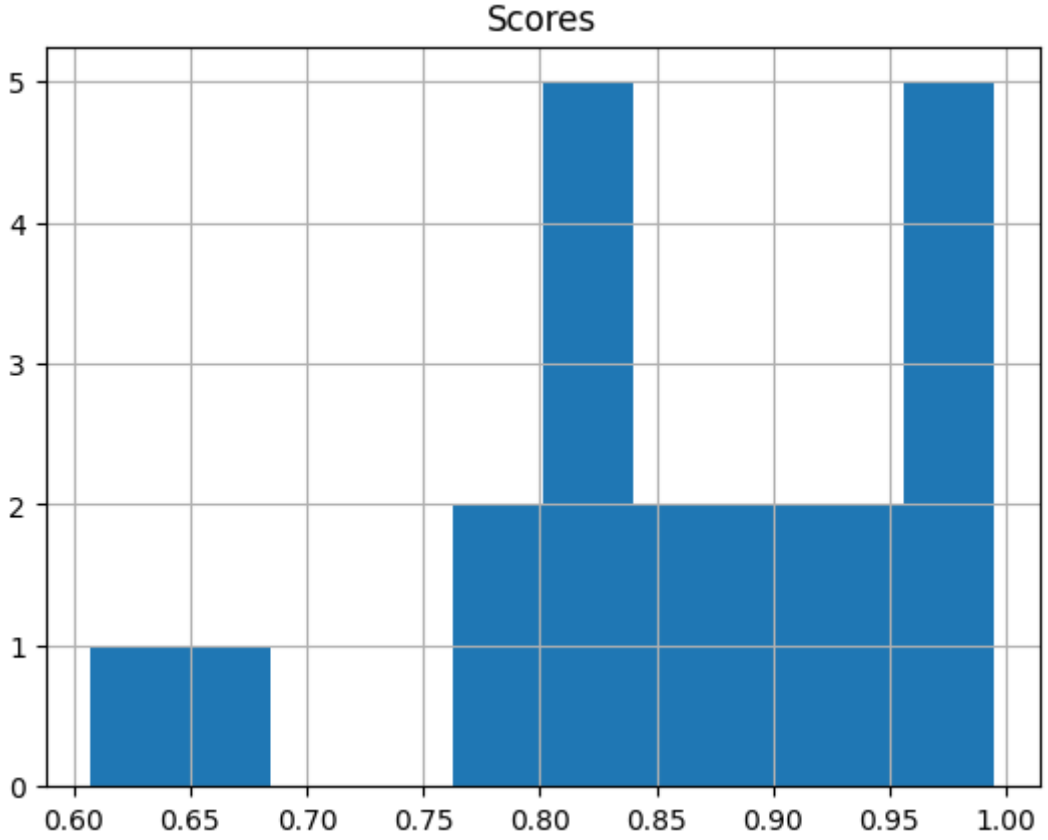

Figura 2 - Histograma de Score das respostas fornecidas pelo modelo

Como pode-se observar pelo histograma, o modelo apresentou um excelente resultado, com apenas 2 respostas abaixo dos 75% de similaridade. Além disso, a similaridade média foi de aproximadamente 86%.

Vale ressaltar que, por muitas vezes, o modelo respondeu de forma totalmente correta, mas a similaridade obtida não foi 100% devido a pequenas variações entre o texto esperado e o obtido, mas que semanticamente são equivalentes. De qualquer forma, é uma boa maneira de se avaliar a assertividade do modelo.

O modelo foi hospedado localmente utilizando-se o Streamlit. Com esse framework, conseguiu-se produzir uma página de layout agradável, como pode-se observar na figura 3.

Figura 3 - Layout da página que hospeda o ChatBot

Como testar o ChatBot

Para testar o ChatBot, basta instalar as bibliotecas importadas no arquivo **chatbot.py** utilizando o comando **pip** e rodar o comando **streamlit run yourdirectory/chatbot.py,** em que yourdirectory é o diretório em que o arquivo foi baixado localmente. Além disso, é preciso se certificar que o arquivo "Procuradoria.pdf" está presente nesse diretório e que a OpenAI key utilizada possui créditos. No código original, a key utilizada está inicialmente com 2.70 dólares de crédito.

Para rodar o código utilizado para a análise, basta instalar as bibliotecas presentes no arquivo **analysis.py** e executá-lo. Porém, é necessário baixar também o treinamento do modelo da biblioteca spacy em português, usando o comando `python3 -m spacy download en_core_web_sm`.

Conclusão

A partir do projeto proposto, foi possível treinar um modelo de ChatBot utilizando RAG e obter resultados satisfatórios, com uma acurácia média de aproximadamente 86%.