Universidade Estadual de Campinas

Depto. de Engenharia de Computação e Automação (DCA)

Faculdade de Eng. Elétrica e de Computação (FEEC)

Processo Seletivo

Iniciação Científica em NLP para Serviços de Saúde

Etapa Prática

Lucas Afonso Carriel RA: 182147 l182147@dac.unicamp.br

Campinas, 2024

1. Introdução

O seguinte relatório apresenta o desenvolvimento de um projeto de programação vinculado à segunda fase (fase prática) do processo seletivo de iniciação científica para a participação no grupo de pesquisa AIMS (Artificial Intelligence for Multimodal Signal Processing), em parceria com a NeuralMind, para o estudo e desenvolvimento de modelos de processamento de linguagem natural em processos da Agência Nacional de Saúde (ANS).

1.1 Motivação

A motivação para participação neste processo seletivo justifica-se por três principais razões, sendo elas:

- Possibilidade de inserção e vivência no mundo acadêmico da pesquisa;
- Aprofundamento no ramo de conhecimento de *deep learning*, em especial no estudo e desenvolvimentos vinculados ao processamento de linguagem;
- Oportunidade de desenvolvimento de pesquisa em parceria com a Agência Nacional de Saúde no ramo de *machine learning*, devido ao aumento contemporâneo na inserção de *scripts,* valendo-se de *IA,* em meu ramo de formação quanto à graduação, a física médica (em especial, na vertente da radioterapia, no processo de formulação do tratamento de pacientes).

1.2 Objetivo

O objetivo do projeto desenvolvido é a criação de um *chatbot*, baseado em *RAG* (Retrieval Augmented Generation), a fim de responder dúvidas sobre o Vestibular Unicamp 2024. O projeto conta com um tratamento inicial de dados disponibilizados pela Resolução GR-031/2023, de 13/07/2023; a aplicação do algoritmo *RAG* para a procura das questões propostas e um *framework* hospedado no ambiente S*treamlit Share* para facilitar a visualização e utilização dele.

2. Metodologia

2.1 ChatGPT – Papel Desempenhado

A ferramenta ChatGPT (versão 2, gratuita) foi amplamente utilizada para o desenvolvimento deste projeto, como recomendado no edital constando os objetivos e métodos de avaliação desta porção do processo seletivo.

Inicialmente, ele foi utilizado para a formulação de um fluxograma, contando com as etapas necessárias para o desenvolvimento do projeto proposto; sendo elas:

- Determinação dos dados utilizados para formar o banco de dados de pesquisa da IA em desenvolvimento;
- Tratamento dos dados para uma procura efetiva do aplicativo;
- Formulação do algoritmo de busca [*RAG*], o qual se baseia em *embeddings* [vide seção 2.3];
- Montagem do *framework* de hospedagem do API para visualização e utilização prática.

No decorrer do projeto, foi-se utilizado o ChatGPT para verificação e consulta quanto a eventuais erros que ocorreram, servindo como mediador para o processo de *debug* do código formulado, em especial o Erro 429 do OpenAI [vide seção 4].

Por fim, ele foi usado para formulação dos *requirements* necessários para o correto funcionamento do API criado em outras máquinas além da utilizada para sua criação [tais requerimentos, assim como as instruções necessárias, podem ser encontradas no repositório compartilhado com os responsáveis pelo processo seletivo].

2.2 Tratamento Inicial de Dados

Os dados brutos que formaram a base do conhecimento da IA utilizada no *chatbot* foram disponibilizados pela Resolução GR-031/2023, de 13/07/2023, a qual consta com todas as informações referentes ao Vestibular Unicamp 2024. No entanto, os dados efetivamente utilizados na codificação são referentes a uma porção do conteúdo, devido ao tempo disponível para a formulação da aplicação [vide seção 4].

A partir da totalidade dos dados utilizados, foram criados *chunks* (porções, subdivisões) do texto, a fim de facilitar a análise de busca; o critério utilizado para sua subdivisão é um dos fatores determinantes de qualidade do código desenvolvido [vide seção 4]. Posteriormente, tais porções foram formulados no formato de *embeddings*, o qual é utilizado pelo algoritmo posterior de busca; tal formato é caracterizado pela impressão dos dados disponíveis em formulações vetoriais [listas de valores] a fim de ser possível estabelecer comparações quantitativas entre termos, valendo-se de algoritmos de cálculo de distância, como explicitado na seção a seguir.

2.3 Algoritmo RAG e OpenAI

A próxima porção do código desenvolvido é constituída pelo algoritmo de buscas e formulações de respostas do *chatbot*.

O algoritmo de busca é baseado em *RAG*, que utiliza os dados em formato de *embeddings* para cálculo de distância (no caso, distância euclidiana) entre vetores para categorização e classificação de proximidade entre termos chaves presentes nas *queries* (consultas, ou perguntas, feitas pelos usuários) com as subdivisões (*chunks*) presentes no banco de dados de referência utilizado. Os critérios utilizados neste algoritmo de busca, também determinísticos na qualidade da aplicação, foram: número de *chunks* utilizados para comparação e formulação da resposta (fator *k*) e o limite de aceitação para uma boa resposta (“distância” mínima entre *queries* e os *embeddings*).

Após a determinação das porções mais relevantes para a formulação da resposta, foi-se utilizada a biblioteca *OpenAI*, que consta com uma inteligência artificial, cuja finalidade é a formulação de uma resposta coerente, baseada nas *k-porções* selecionadas (de acordo com o algoritmo *RAG*); para tanto, foi utilizada uma *API-Key* disponibilizada pela plataforma *OpenAI*, a qual foi um grande fator de dificuldade enfrentado na formulação do projeto [vide seção 4]. O resultado da sua aplicação é a resposta exposta no *framework* utilizado.

2.4 Framework de Hospedagem

Por fim, a aplicação foi hospedada no ambiente *Streamlit Share*, ambiente seguro e gratuito que conta com as funcionalidades necessárias para a execução do *chatbot*: presença de uma interface para inserção das *queries* e da visualização da resposta proposta pela IA; assim como métodos de verificação e mensagens de alerta em casos de erros na sua execução.

3. Resultados

O resultado do trabalho desenvolvido foi um *chatbot* cunhado como CDVU-2024 (Centro de Dúvidas do Vestibular Unicamp 2024), presente em um diretório na plataforma Github, já compartilhado por e-mail com os responsáveis pelo processo seletivo da iniciação científica, constando com uma série de documentos:

- *Arquivo ".env"* – arquivo que consta com a chave API utilizada para execução da aplicação (sendo sua atualização necessária a depender do usuário que a utilize);
- *Arquivo "Dados.md"* – arquivo com os dados extraídos pela Resolução indicada, sendo o seu formato (Markdown) utilizado para formulação dos *chunks* em *embeddings*;
- *Arquivo "README"* – consta com as instruções necessárias para que a aplicação possa ser executada na máquina do usuário;
- *'Relatório'* – arquivo que dispõe deste relatório desenvolvido a fim de explicitar os meios utilizados para a elaboração do projeto;
- *Arquivo "project_final.*py" – correspondente ao código bruto da aplicação, o qual apresenta comentários em todo o seu desenvolvimento;
- *Arquivo "requirements.*txt" – arquivo de texto que consta com as bibliotecas, e suas versões, utilizadas na formulação da aplicação.

4. Comentários Finais

4.1 Aprimoramentos Possíveis

O tempo limitado para a formulação da aplicação reflete quanto ao resultado entregue, uma vez que os principais fatores que determinam a sua qualidade são:

- *Banco de dados*: o arquivo “Dados.md”, como explicitado, conta com os dados utilizados nas porções posteriores da aplicação; sendo, pois, a sua base. Assim, o seu polimento, quanto ao conteúdo presente e a forma como é apresentado, melhorariam a qualidade do *chatbot* desenvolvido;
- *Parâmetros na criação de embeddings:* em um maior intervalo de tempo, poderiam ser testadas as efetividades de diferentes valores quanto ao tamanho dos *chunks* utilizados, o tamanho da sua sobreposição e o critério de subdivisão, a fim de determinar valores ótimos para a aplicação;
- *Parâmetros no algoritmo RAG*: o fator *k* teve seu valor escolhido com base nas referências utilizadas [vide seção 5]; no entanto, em um maior intervalo de tempo, o seu teste, em paralelo com os parâmetros dos *embeddings*, permitiria a determinação do seu valor ideal.

Apesar de não ser um fator determinístico, a qualidade da interface na plataforma *Stremalit Share*, poderia ser aprimorada, a fim de aumentar a recepção de novos usuários à plataforma, no que se refere à qualidade do fundo do ambiente, as fontes utilizadas e efeitos de busca de perguntas, assim como maior acesso ao usuário do banco de dados utilizado através da interface.

4.2 Erro 429 da OpenAI – Principal Dificuldade Enfrentada

No decorrer do projeto, foram encontrados empecilhos quanto ao código: erros vinculados à sintaxe, indexação ou modulação nos elementos presentes, porção comum no processo de codificação; os quais foram resolvidos com uma série de pesquisas, em fóruns, documentações de bibliotecas e módulos específicos utilizados [vide seção 5], e com a ajuda da plataforma ChatGPT.

No entanto, o principal fator de erro presente no decorrer deste projeto foi o Erro 429 da plataforma OpenAI, referente à limitação de quotas, presentes para usuários de acesso gratuito, em testes diários que poderiam ser realizados, vinculados à API-Key disponibilizada para cada perfil pela plataforma. Este erro consumiu a maior parte do processo de *debugging* do código em questão, tendo sido resolvido com a medida de compra de créditos na plataforma, a fim de que tais quotas fossem expandidas.

Apesar da orientação de não ser necessário o uso de investimentos monetários pessoais no desenvolvimento desta aplicação, em decorrência da presença de modelos alternativos ao OpenAI, como o modelo llama 3 de 70B de parâmetros; esta medida de investimento foi escolhida por mim devido à:

- O único erro presente no código desenvolvido, até o dia 09/06/2024, às 14:00, foi o Erro 429 vinculado à API-Key utilizada; sendo, pois, o único obstáculo para a verificação da capacidade de execução da minha aplicação;
- Não foi optado por mim o uso do modelo llama 3, devido à sua complexidade de aplicação, que está além dos meus conhecimentos atuais na área de programação.

No entanto, deixo explicitado neste relatório que estou disposto e ansioso para aprender a valer-me de modelos, como o llama 3, em desenvolvimentos futuros de códigos de tratamento de linguagem, como consta no escopo da Iniciação Científica proposta. Desde este momento, agradeço pela oportunidade e pela compressão.

5. Referências

- Sintaxe de arquivos Markdown – [Link]
- Diretório Github: biblioteca OpenAI – [Link]
- Diretório Github: modelo de domínio público utilizado como base no desenvolvimento do chatbot – [Link]
- Streamlit documentation – [Link]
- Dotenv documentation – [Link]