Relatório Exercício de seleção curso deep learning

Para a recuperação de informações (IR) que é o que o exercício se propõe, foi feita a sugestão de utilizar duas arquiteturas: BM25 e GPT-3.

Tentando entendê-las mais a fundo, o BM25 é de fato uma boa escolha para o exercício. Mas, o GPT-3 por se tratar de um modelo de gerar textos, responder perguntas como de fato é o ChatGPT, esse tipo de arquitetura não seria a melhor escolha para IR.

Logo, o exercício desenvolvido no Google Colab foi somente utilizando o BM25 para a recuperação de informação. E, utilizando o ChatGPT para eventuais dúvidas que surgiriam, assim utilizando o modelo GPT-3.

Para utilização da CISI collection, foi pesquisado o que cada arquivo dentro do zip significaria. Portanto, foram utilizados somente 3 arquivos.

O .ALL que tem todos os resumos - logo o corpus utilizado no modelo.

O .QRY que tem as buscas por resumos.

O .REL que seria a devolutiva das buscas relativas a cada resumo - ground truth.

Todos os 3 arquivos foram necessários tratamentos. Sempre deixando as palavras numa lista de listas. Ou seja, cada primeira posição seria o primeiro artigo/consulta e nessa primeira posição teria outra lista com as palavras separadas. Foi removido stopwords e reduzir as palavras ao seu radical também. Para esse tipo de tratamento foi utilizado a biblioteca NLTK.

Remover stopwords ("what", "is", "that"...) ajuda a não utilizá-las para a recuperação de informações. Ou seja, se uma pesquisa (.QRY) utiliza muito a palavra "what" o modelo iria pegar o resumo que mais tem a palavra "what" não sendo necessariamente do mesmo assunto, logo fazendo a ligação de pesquisa com resumo de maneira equivocada. Por isso a remoção desses tipos de palavras para ajudar na busca e deixá-la mais assertiva.

Também utilizado o stemmer que seria para reduzir as palavras ao seu radical, assim diminuindo a quantidade de palavras que há no vocabulário (implícito à biblioteca Gensim - tratado mais a frente) e ao cálculo de IDF, logo assim palavras semelhantes não seriam tratadas como palavras raras, melhorando a performance do modelo também.

Os tratamentos acima foram sugeridos pelo chatGPT para a melhor performance do modelo.

Depois desses tratamentos acima descritos, também foi pesquisado no chatGPT como utilizaria o modelo BM25. E foi sugerido uma biblioteca Gensim ("from gensim.summarization.bm25 import BM25"). Nessa biblioteca se torna mais fácil a implementação do modelo, pois os cálculos de IR são abstraídos do código.

Logo, apenas um comando: "model_bm25 = BM25(corpus)" já faz todos os cálculos de maneira otimizada, deixando assim o modelo já treinado para o input das pesquisas.

Também foi necessário calcular o IDF, como mencionado acima, que faz o cálculo da raridade das palavras que apareceram no corpus. Sendo assim mais fácil de ligar o *corpus* às consultas que tem a mesma palavra (ou radical da palavra). O pacote do Gensim já tem esse atributo, também ajudando nos cálculos dos vocabulários, e na otimização do código.

Dessa maneira foi feito o ranqueamento dos artigos relacionados à cada pesquisa. Logo, para cada pesquisa teríamos a mesma quantidade de documentos, mas ordenados de maneira distinta, seguindo o score de probabilidade daquele documento pertencer àquela pesquisa.

Portanto, para calcular a assertividade do modelo ser tornaria um pouco mais complicado, pois a relação de artigos/documentos relacionados a cada busca tem uma quantidade variável.

Logo, foi adotado que cada busca deveria retornar a mesma quantidade de documentos que o arquivo .REL para assim facilitar sua comparação e o cálculo de assertividade. Mas, não necessariamente seria a melhor maneira de tratar a acurácia do modelo.

Poderia limitar em 10 documentos por pesquisa, ou qualquer outro número; poderia também fazer um corte (mediana) no score e trazer somente os documentos que estão acima desse corte (score). Essas maneiras dificultariam um pouco o cálculo pois nem sempre o denominador da razão entre acerto e quantidade de documentos seria o mesmo. Portanto, foi decidido seguir pelo cálculo exposto da primeira maneira.

Existem também consultas vazias, também houve esse tratamento no cálculo da assertividade do modelo. Atribuindo o valor de -9 para a consulta vazia. Assim, quando foi calculado a assertividade foi desconsiderado esses casos.

Dessa maneira, a assertividade/precisão do modelo ficou por volta de 24%. Particularmente o número não foi próximo do que era esperado, mas devido às consultas no chatGPT e até mesmo no documento de implementação do Gensim não foi encontrada outra maneira de conseguir melhorar esse número de assertividade, dado o tempo escasso também.

Ficando assim como próximos passos: utilizar maneiras diferentes de medir a assertividade/acurácia do modelo, e consultar possíveis melhorias nos parâmetros do modelo para melhorar a acurácia, tanto na documentação do Gensim quanto no chatGPT.