Documentação Research Challenge – Recuperação de Informação

Francisco Teixeira Rocha Aragão - 2021031726
Lorenzo Carneiro Magalhães - 2021031505
franciscoaragao@protonmail.com
lorenzocarneirobr@gmail.com
Departamento de Ciência da Computação - UFMG
Belo Horizonte, Minas Gerais, Brasil

Abstract

Máquinas de busca modernas, além de lidarem com a tarefa de indexar um enorme corpus de documentos, precisam também retornar rankings de qualidade para as queries do usuário. Desse modo, o presente trabalho foca nessa tarefa de produzir rankings de qualidade utilizando diferentes técnicas vistas na disciplina.

CCS Concepts

• **Information systems** → **Information retrieval**.

Keywords

ranking, recuperação, informação

ACM Reference Format:
Francisco Teixeira Rocha Aragão - 2021031726 and Lorenzo Carneiro Magalhães - 2021031505. 2018. Documentação Research Challenge – Recuperação de Informação. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Recuperação de Informação - RC).* ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introdução

O trabalho de pesquisa proposto na disciplina de recuperação de informação implementa um sistema que, dado um corpus de documentos e as queries do usuário, retorna um ranking de documentos relevantes ao usuário. Como a execução do trabalo está ligada a uma competição, o objetivo é retornar rankings com melhores resultados de nDCG. Dessa forma, as próximas seções explicam as estratégias utilizadas em cada submissão.

2 Primeira submissão

Identificação: submission.csv

Score: 0.37940

A primeira submissão foi feita seguindo abordagens simples vistas na disciplina, envolvendo preprocessamento dos textos, indexação e ranqueamento. Primeiramente, o corpus é lido e preprocessado com algumas funções simples, como aplicar *lower case*, tokenização, remoção de stopwords e de tokens muito curtos (como 1 caractere). Dessa forma, é salvo internamente como estruturas de índice a lista de IDs de cada entidade, além dos tokens de cada documento para serem utilizados durante o ranqueamento. Assim, durante a leitura das queries, cada uma delas é preprocessada para então o algoritmo de Bm25 ser utilizado. Com isso, os scores são salvos e retornados posteriormente.

Vale destacar que essa submissão foi bem simples e utilizou conceitos clássicos da matéria, sem realizar qualquer tipo de treinamento. Foram utilizadas as bibliotecas nltk para preprocessamento, além de rank_bm25 para realizar o ranqueamento com o algoritmo Bm25.

3 Segunda submissão

Identificação: submission.csv

Score: 0.50639

A segunda submissão tentou explorar técnicas um pouco mais avançadas dado o resultado pouco satisfatório do primeiro método. Dessa forma, agora foram utilizadas estratégias mais robustas, como o método rm3 e cross encoder para reranking.

Assim, foi-se utilizado a abordagem com Rm3, que é um modelo para expansão de consultas. Como visto na disciplina, isso é bastante útil para aproximar a query dos documentos relevantes no espaço de documentos, além de facilitar o matching de vocabulário (o que acaba auxiliando o bm25). Ademais, também foi adotada a estratégia de reranking, com o modelo MsMarco sendo utilizado, que é um modelo treinado com linguagem natural focado em tarefas de recuperação de informação e question answering. Com sua abordagem de Cross Encoder, o modelo recebe tanto a query quanto textos dos documentos como input, resultando em um bom entendimento sobre todas as informações. Vale destacar que como essa abordagem utiliza a query e os documentos, é necessária de ser feita *online*, o que consome bastante tempo, porém não foi algo relevante na tarefa visto que não tinhamos essa restrição, além de usarmos um modelo mini para maior eficiência.

Desse modo, o processo foi feito da seguinte forma. Inicialmente, como estamos trabalhando com métodos mais robustos, a etapa de preprocessamento não foi feita para essa submissão. Assim, inicialmente os modelos são carregados e seus parâmetros definidos (bm25 com k1 = 0.92 e b=0.36, que foram utilizados pensando que o corpus não tem documentos muito longos, além de rm3 com até 10 termos adicionados nas consultas, sendo utilizados 50 documentos para coleta de informações para expansão).

Vale destacar que os modelos foram carregados sobre os arquivos de indíce gerados sobre o utilitário Anserini. Como a biblioteca Pyserini foi utilizada para trabalhar com os modelos (rm3 e bm25), o formato do corpus deveria ser padronizado para a estrutura id: ..., contents: .... Assim, após esse processamento feito, a biblioteca Pyserini é chamada para relaizar a criação do indíce no formato

Recuperação de Informação - RC, Junho 23, 2025, Belo Horizonte, MG

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

correto. Com isso, acaba que a biblioteca que cuida de toda a tarefa de indexação, não sendo necessário maiores trabalhos manuais nessa parte.

Após esse processamento inicial e padronização do corpus em um indice, o ranqueamento começa. O processo é bem simples no geral, o bm25 é executado sobre a query, retornando K melhores documentos. Em sequência, o modelo de reranking é utilizado sobre essa amostra para então retornar os 100 melhores resultados para a query. Por fim, os resultados são interpolados a partir de um valor alpha, para então serem salvos e submetidos na competição.

As bibliotecas utilizadas foram a pyserini para cuidar do corpus e dos métodos clássicos, além de sentence_transformers para trabalhar com o modelo de cross encoder. O valor de K testado foi de 1000 documentos e alpha como 0.7 (dando mais peso para o modelo de reranking do que o bm25 + rm3). É válido comentar como essa abordagem mostrou-se muito melhor na prática do que utilizar apenas o bm25. Como muitas variáveis foram alteradas, não da para dizer de momento o que foi mais impactante, se foi a nova estrutura de índice, o modelo de reranking ou a expansão de queries, porém toda essa junção de técnicas ajudou na melhoria dos resultados.

4 Terceira submissão

Identificação: submission3.csv

Score: 0.49218

A terceira submissão seguiu a mesma lógica da segunda, porém teve apenas alguns parâmetros alterados para testar diferentes ideias. Primeiramente, foi feito o preprocessamento do texto de modo semelhante ao que foi explicado na primeira submissão (já que esse processo não foi realizado na segunda submissão). Além disso, os parâmetros do Bm25 foram alterados para valores padrão, com k1 = 1.2 e b = 0.75. Ademais, o número de documentos para o reranking foi reduzido para apenas 500 (o que melhorou a performance) e o peso do modelo foi alterado para 80%. Nenhuma mudança foi feita na lógica do processo, somente a alteração dos parâmetros utilizados. Percebe-se que não houve muita diferença na prática, mesmo que houve uma piora dos resultados, não foi também algo muito expressivo.

5 Quarta submissão

Identificação: submission3.csv

Score: 0.48224

Agora na quarta submissão, seguindo a mesma ideia das submissões anteriores, nada foi alterado em relação a lógica de execução da tarefa. A única modificação foi a utilização do score completo do modelo MsMarco para realizar o ranking dos 100 melhores documentos. O bm25 ainda é utilizado para gerar os melhores 1000 documentos, porém o score final é definido somente pela LLM. Isso foi feito para observar o quão bom ficaria o modelo mais robusto sozinho, sendo um baseline de comparação com a terceira submissão. Ao observar os resultados, foi possível notar um pior score em comparação a submissão anterior, mostrando que embora o uso do modelo LLM seja útil, não pode ser utilizado cegamente para a tarefa.

6 Ideias não testadas

Por fim, vale destacar que outras estratégias foram testadas localmente, porém sem sucesso. Como exemplo, abordagens envolvendo modelos doc2query foram testass, em que um trecho do documento é passado ao modelo e então é gerado um query que o representa, facilitando assim a etapa de matching com as queries. Entretanto, por motivos práticos, essa abordagem não seguiu adiante, sendo necessário maior poder computacional para cumprir a tarefa, que iria demorar muito em um ambiente computacional sem GPU. O mesmo problema também afeta abordagens learning to rank em que ocorre o aprendizado do modelo, que dependem de maior poder computacional para terem sucesso.

7 Execução

Todo o código foi sumarizado em um notebook Jupyter, sendo separados para cada submissão feita. Vale destacar que é necessário fazer algumas etapas de instalação conforme mencionado no decorrer do documento, além das bibliotecas presentes no arquivo "requirements" enviado junto ao código. Vale destacar que é necessário a instalação do Java para correto funcionamento do código.

8 Conclusão

Percebe-se ao fim da tarefa, como o problema é complicado e necessita de diferentes abordagens para ser solucionado com eficiência. Vale destacar que embora diferentes métodos tenham sido testados para cumprir a tarefa, novas abordagens ainda podem ser utilizadas, como exemplo, a utilização de enriquecimento das infromações dos documentos com outros LLMs, ou utilização de novos métodos e heurísticas para ranquear as informações.

9 Referências

Escolha do valor de K1 e B do BM25

Informações do modelo MsMarco e documentação da api

Lista de resultados em diferentes modelos de recuperação de informação